Basics of Matrix Algebra with R

(Notes to accompany MAS6011/MAS465 & MAS6003/MAS473/MAS371)

Dr Nick Fieller

Department of Probability & Statistics

University of Sheffield

2010/11

Contents

Basic Matrix Algebra

0 Introduction

0.0 Books

Abadir, Karim M. & Magnus, Jan R. (2005) ***Matrix Algebra*** Cambridge University Press.

Searle, Shayle, (2006) ***Matrix Algebra Useful in Statistics*** Wiley-InterScience.

Harville, David A. (2008) ***Matrix Algebra from a Statistician's Perspective*** Springer-Verlag New York

Basilevsky, Alexander (2005) ***Applied Matrix Algebra in the Statistical Sciences*** (Dover)

There are many books on Matrix Algebra and the one listed above by Abadir & Magnus is not essentially different from many others; it just happens to be the one used for writing much of the material in this set of notes and exercises. In particular many of the exercises are taken from this book. Some of the exercises and examples are taken from Searle and also Basilevsky, Harville has been consulted for ideas. None of these give any guidance on the computational aspects using **R.** It is not expected that these books will need to be consulted since the notes are intended to be an expanded synthesis of those parts of these books most relevant to the need of those embarking upon the MSc in Statistics in Sheffield.

As well as describing the basics of matrix algebra, including numerical calculations 'by hand', for example of matrix multiplication and inversion, the notes give guidance on how to do numerical calculations in **R.**

R is an open source system and is available free. It is 'not unlike' the expensive commercial package S-PLUS, the prime difference is that **R** is command-line driven without the standard menus and dialog boxes for statistical operations in S-PLUS. Otherwise, most code written for the two systems is interchangeable. There are however a few differences, for example in the way external files are referenced (S-Plus uses a single / and **R** uses a double // in full file pathnames). There may also be differences in the available optional arguments. These are quickly verified by use of the Help system.

The sites from which **R** and associated software (extensions and libraries) and manuals can be found are listed at

http://www.ci.tuwien.ac.at/R/mirrors.html

The nearest one is at

http://cran.uk.r-project.org (in Bristol, UK)

Free versions of full manuals for **R** (mostly in PDF format) can be found at any of these mirror sites. There is also a wealth of contributed documentation.

0.1 Objectives

The aim of these notes is to provide a guide to elementary matrix algebra sufficient for undertaking the courses on Multivariate Data Analysis and Linear Models. In fact the notes go a little further than this, providing an initial guide to more advanced topics such as generalized inverses of singular matrices and manipulation of partitioned matrices. This is to provide a first step for those who need to go a little further than just the MSc courses on Multivariate Data Analysis and Linear Models, for example when embarking on the dissertation. Sections which are more advanced but nevertheless are very useful to know for complete understanding of the courses are marked with a **+** in the numbering, e.g. **1.8⁺ Partitioned Matrices.** Particularly advanced sections going well beyond that necessary for the courses are indicated with a ★★ in the numbering, e.g. **7.3★★ Moore–Penrose Inverse.** It is to be understood that sub-sections of those marked with a **+** or ★★ are also similarly categorised.

Since the applications envisaged for the majority of users of these notes are within Statistics most emphasis is given to ***real*** matrices and mostly to real ***symmetric*** matrices. Those who want to go further into properties of complex and asymmetric matrix algebra will find a comprehensive introduction in Abadir & Magnus. *Unless specifically indicated, all matrices are presumed to be **real.***

In addition to the algebraic manipulation of matrices the notes give numerical examples since it is necessary to be able to do numerical calculations, both by hand and by using **R.** Much of the exposition is by presenting examples with solutions. There are additionally some examples for self study and solutions are available separately (for the desperate) to these but for numerical questions the calculations are given only in **R** and in some cases only the numerical answers are given. A brief guide to installing and running **R** is included.

0.2 Guide to Notation

- Generally, matrices are denoted by uppercase letters at the beginning and end of the alphabet: A, B, C, D, U, V, W, X, Y, Z
- Generally [column] vectors are denoted by lowercase letters at the beginning and end of the alphabet: a, b, c, d, u, v, w, x, y, z
- Generally *elements* of a vector x are denoted by $x_1, x_2, \ldots, x_p$
- Generally *elements* of a matrix X are denoted by $x_{11}, x_{12}, \ldots$.
- Sometimes the *columns* of a matrix X are denoted by $x_1, x_2, \ldots, x_q$
- Usually published texts indicate matrices and vectors by bold fonts or underscores but this is not be done here except in cases of need of resolving ambiguity, clarity and emphasis: **A, B, x, y,** $\underline{a}$, $\underline{x}$, $\underline{\underline{B}}$,……
- Lower case letters in the middle of the alphabet i, j, k, l, m, n, p, q, r, s, t generally indicate integers. Often i, j, k, l are used for dummy or indexing integers (e.g. in summation signs) whilst m, n, p, q, r, s, t are usually used for fixed integers, e.g. for i = 1, 2, …, n.
- The upper case letters H, I, J are usually be reserved for special matrices.
- The transpose of a matrix X is indicated by X′ . Some texts may use an alternative X^T and this may be used in some circumstances here.
- The inverse of a [non-singular] matrix X is indicated as X^{-1}.
- The [Moore-Penrose] generalized inverse of a [not necessarily non-singular or square] matrix is indicated as A^+ and the generalized inverse by A^- (A^+ is a restricted form of A^-).
- There are some exceptions to these conventions.

0.3 An Outline Guide to R

0.3.1 What is R?

R is a powerful interactive computer package that is orientated towards statistical applications. It will run on the most commonly used platforms (or operating systems) Windows, Linux and Mac. The notes here are orientated towards use on a Windows platform. It consists of a *base system* that can be downloaded without charge together with many contributed packages for specialist analyses. It offers:–

- an extensive and coherent set tools for statistics and data analysis
- a language for expressing statistical models and tools for using linear and non-linear statistical models
- comprehensive facilities for performing matrix calculations and manipulations, enabling concise efficient analyses of many applications in multivariate data analysis and linear models
- graphical facilities for interactive data analysis and display
- an object-orientated programming language that can easily be extended
- an expanding set of publicly available libraries or packages of routines for special analyses
- libraries or packages available from the official Contributed Packages webpages are thoroughly tested by the **R** Core Development Team
- packages have manuals, help systems and usually include illustrative datasets

0.3.2 Installing R

Full instructions for installing **R** are given on the **R** Project home page at http://www.r-project.org/. The first step is to choose a site geographically close to you from which to download the package. Next choose the appropriate operating system, select the option base and download the system. Accepting the option to run the download file will install the package after downloading it. Accepting the default options for locations etc is simplest but these can be customized. By default an icon (ℝ [with version number]) is placed on the desktop. Clicking on the icon will open an **R** session (i.e. start the **R** program). The **R** Graphical user interface (RGui) opens and you are presented with a screen like this:

```
R version 2.9.0 (2009-04-17)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

0.3.3 R is an interactive program

The symbol **>** is the *command line prompt symbol;* typing a command or instruction will cause it to be executed or performed immediately. Along the top of the window is a limited set of menus. The `Packages` menu allows you to install specific packages (which needs to be done only once) and then to load them into the session. Each time a new session is started you need to load the packages which you will need. This can be done from the `Packages>Load Packages...` menu or by the command `library(packagename)`. Some of the commands needed for matrix manipulations are within the `MASS` library which is automatically installed (together with a few others such as `stats, Matrix, graphics, ...`) when **R** is first installed, i.e. it does not need to be installed from the `Packages` menu but it does need to be loaded if needed during each R session. [MASS is *Modern Applied Statistics with S*, by W N Venables & B D Ripley, (2002) Springer]. Some packages are automatically loaded during each **R** session (e.g. `stats` and `graphics` but not `Matrix` and `MASS`). To discover which packages are loaded during a session issue the command `search()`.

A convenient way of running several commands in sequence is to open a *script window* using the `File>New script` menu which opens a simple text editing window for typing the commands. Highlighting a selection and then clicking on an icon in the menu bar will run the commands in the selection. Lines in the script window can be edited and run again. A script can be saved in a file (with default extension .R) and opened in a later session via the menus.

0.3.4 R is a function language

All commands in **R** are regarded as *functions*, they operate on *arguments*, e.g. `plot(x, y)` plots the vector x against the vector y — that is it produces a scatter plot of x *vs.* y. Even Help is regarded as a function:— to obtain help on the function `matrix` use `help(matrix)`. To end a session in **R** use `quit()`, or `q()`, i.e. the function `quit` or `q` with a null argument. In fact the function `quit` can take optional arguments, type `help(quit)` to find out what the possibilities are.

0.3.5 R is an object orientated language

All entities (or 'things') in **R** are ***objects***. This includes vectors, matrices, data arrays, graphs, functions, and **the results of an analysis.** For example, the set of results from performing a two-sample t-test is regarded as a complete single object. The object can be displayed by typing its name or it can be summarized by the function `summary()`. Even the results of `help` are objects, e.g. of `help(matrix)`. If you want to store the object created by this for later examination (though the need for this may be rare), giving it say the name `matrixhelp` then do `matrixhelp<-help(matrix).` Typing `matrixhelp` will print the help information on the screen (or it can be exported).

0.3.6 Saving objects and workspaces

Objects such as a matrices and vectors (see below) created during an **R** session can be saved in an **R** workspace file through the `File>Save Workspace...` menu or via the icon. They can be loaded into the **R** session by the menu or (if the default RData file extension is accepted) the file can be located in Windows Explorer and clicking initiates an **R** session with this workspace open. Issuing the command `objects()` (or equivalently `ls()` or use the menu under **`Misc`**) will list the objects created (or retrieved from a workspace) during the current session.

When you close down **R** you are prompted to whether you want to save the workspace image. If you have loaded a workspace during the session then this will be overwritten by the current one. When you next run **R** you will start with an empty worksheet and can load any previously saved one.

0.3.6.1 Mistakenly saved workspace

BEWARE:– If you have created all the objects in the workspace during the current session (i.e. you have not loaded or opened a previously saved workspace) when you accept the invitation "Save workspace image?" at the end of a session it will be saved somewhere on your drive. ***When you next start R this workspace will be automatically restored***. This can have unexpected consequences and can cause mysterious problems because you will have objects in the workspace that you might not have intended to be there. To cure it you can remove all objects by `rm(list = ls(all = T))` or use the menu under **`Misc`** and choose `Remove all objects`. This should give a response `character(0)` to the command `ls()`.

0.3.7 R is a *case-sensitive* language

Note that **R** treats lowercase and uppercase letters as different, for example inverting a matrix is performed using the function `solve()` but **R** does not recognize `Solve()`, nor `SOLVE()`, nor…… The objects `x` and `X` are distinct (and easy to confuse). The function `matrix` and the library `Matrix` are distinct.

0.3.8 Obtaining help in R

Obtaining help on specific functions during an **R** session can by done by using either `help(functionname)` or `?functionname`. This will give you the list of possible arguments and the list of possible values produced. There may also be examples of their use, including script files which can be cut&pasted into a a script window for you to run. Typing `example(functionname)` may run one or more examples. This of course requires you to know the name of the function.

Typing `library(help=libraryname)` will give summary description of the library together with an index of all functions and datasets supplied with the library. Having found the name of a function or a dataset then use `help(.)`, `?` and `example(.)` to find out more about it. For example `library(help=stats)`lists all the functions in library `stats`; these are mostly statistical functions such as `t.test` and then `help(t.test)` shows exactly how to perform a Student's t-test.

To find out what various packages can do look at the CRAN website and click on packages. This has a basic search facility with CTRL+F (for the Windows Find dialogue box) which will search for an exact match for the character string entered in the dialogue box. For example to find packages which have functions for imputation of missing values then go to the Packages page on the CRAN project page and scroll down to the list headed **`Available Bundles and Packages,`** press CTRL+F and enter `impute` in the dialogue box. This will list in turn `arrayImpute`, `impute`, `imputeMDR`, and `yaImpute`. This technique will only find strings which appear on the very brief summary descriptions of the packages. A more thorough search can be performed from within an **R** session with `help.search` or `??`

For example `help.search(”characterstring”)` or equivalently, `??characterstring`, will search all installed packages for an approximate match in the summary description of each function in the package to the `characterstring`. The default is to use fuzzy matching which may have unexpected consequences. For example using `help.search(”impute”)` or equivalently `??impute` will also find all such occurrences of `compute`. To find out how to avoid this and instead use exact matching try `help(help.search)`.

To find out more about **R** the **R** website has links (under `Documentation`) to various manuals and other contributed documentation. Following the link from the CRAN page under `Other` and **R-related projects** gives to **The R Wiki** **at**

http://wiki.r-project.org/rwiki/doku.php

0.4 Inputting data to R

0.4.1 Reading data from the keyboard

Small amounts of data can be typed directly from the keyboard. For example to create a vector `x` of length 4 containing the four numbers 1.37, 1.63, 1.73, 1.36 do `x<-c(1.37, 1.63, 1.73, 1.36)`and to enter numbers into a matrix see the next section §0.5. The function `scan()` can be used to enter data and will stop when a complete blank line is read. For example:–

```
> x<-scan()
1: 1.37
2: 1.63 1.73
4: 1.36
5:
Read 4 items
> x
[1] 1.37 1.63 1.73 1.36
>
```

`scan()` is a very flexible function with facilities for entering tables of numbers etc, to find out more type `help(scan)`.

0.4.2 Reading data from files

The three main functions for reading tabular data from files are, `read.table()`, `read.csv()`and `read.delim()`. The first is used primarily for plan text files (i.e. with extension .txt or .dat), the second for comma separated values (e.g. as produced by Excel) and the third for tab separated values. The default format of the data in `read.table()`is that the first row should contain the column names (i.e. variable names) and the first item of each row is the row name, so the first row contains one fewer items than the other rows. If the data are not in such a standard form then look at the help system to find out how to use the additional arguments `header`, `row.names` and `col.names` to read the data correctly.

If a data file has been saved during an **R** session (using the `save()` function (see `help(save)`) then the data can just be retrieved by `load("filename")`.

The `source("filename")` function will execute all the **R** commands in the specified file and this can be a convenient method of delivering a data file.

The library `foreign` can be used for reading data files created by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ..., (but not S-PLUS). There are commercially available packages for reading S-PLUS and other files and with a substantial discount for academic use.

0.5 Guide to Matrix Operators in R

The operations given below assume that the orders of the matrices involved allow the operation to be evaluated. More details of these operations will be given in the notes later.

- R is ***case sensitive*** so A and a denote distinct objects
- To control the number of digits printed to 3
  `options(digits=3)`
- Creating a vector x
  `x<-c(x₁,x₂,...,xₚ)`
- To access an individual element in a vector x, the i^{th},
  `x[i]`
- Creating a matrix A:
  `A<-matrix(data, nrow=m, ncol=n, byrow=F)`
- To access an individual element in a matrix A, the $(i,j)^{th}$,
  `A[i,j]`
- To access an individual row in a matrix A, the i^{th} ,
  `A[i,]`
- To access an individual column in a matrix A, the j^{th},
  `A[,j]`
- To access a subset of rows in a matrix A
  `A[i₁:i₂,]`
- To access a subset of columns in a matrix A
  `A[,j₁:j₂]`
- To access a sub-matrix of A
  `A[i₁:i₂,j₁:j₂]`

- Addition A + B: `A+B`
- Subtraction A – B: `A–B`
- Multiplication AB: `A%*%B`
- Transpose A′: `t(A)`
- Inversion A^{-1}: `solve(A)`
- Moore Penrose generalized inverse A^{+}: `ginv(A)`(in `MASS` library) [or `MPinv(A)`(in `gnm` library)]
- Note: `ginv(.)` will work with almost any matrix but it is safer to use `solve(.)` if you expect the matrix to be non-singular since `solve(.)` will give an error message if the matrix is singular or non-square but `ginv(.)` will not.
- Determinant det(A) ***or*** |A|: `det(A)`
- Eigenanalysis: `eigen(A)`
- Singular value decomposition: `svd(A)`
- Extracting a diagonal of a matrix
  `diag(A)`
- Trace of a matrix
  `sum(diag(A))`
- Creating a diagonal matrix
  `diag(c(`x_{11}, x_{22}, …x_{pp}`))`
- Creating a diagonal matrix from another matrix
  `diag(diag(A))`
- Changing a data frame into a matrix:
  `data.matrix(dataframe)`
- Changing some other object into a matrix
  `as.matrix(object)`

- To join vectors into a matrix as columns
  `cbind(vec1,vec2,...,vecn)` — $\texttt{cbind(vec}_1\texttt{,vec}_2\texttt{,...,vec}_n\texttt{)}$
- To join vectors into a matrix as rows
  $\texttt{rbind(vec}_1\texttt{,vec}_2\texttt{,...,vec}_n\texttt{)}$
- To join matrices A and B together side by side: `cbind(A,B)`
- To stack A and B together on top of each other: `rbind(A,B)`
- finding the length of a vector
  `length(x)`
- finding the dimensions of a matrix
  `dim(A)`

0.5.1 Examples of R commands

This section is for quick reference, details are explained later in the text.

0.5.1.1 Expressions

$x(X'X)^{-1}x'$: `x%*%solve(t(X)%*%X))%*%t(x)`

0.5.1.2 Inputting data

```
> A<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=F)
> B<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> A[1,2]
[1] 3
> A[1,]
[1] 1 3 5
> A[,2]
[1] 3 4
```

```
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> B[2,2]
[1] 5
> B[2,]
[1] 4 5 6
> B[,3]
[1] 3 6
```

```
> C<-matrix(c(1,2,3,4,5,6),2,3)
> D<-matrix(c(1,2,3,4,5,6),2,3,byrow=T)
> C
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> D
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

0.5.1.3 Calculations

```
> A+B
     [,1] [,2] [,3]
[1,]    2    5    8
[2,]    6    9   12
```

```
> A-B
     [,1] [,2] [,3]
[1,]    0    1    2
[2,]   -2   -1    0
```

```
> 2*A
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    4    8   12
```

```
> A*2
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    4    8   12
```

Beware:

```
> A%*%B
Error in A %*% B : non-conformable arguments
> t(A)
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> t(A)%*%B
     [,1] [,2] [,3]
[1,]    9   12   15
[2,]   19   26   33
[3,]   29   40   51
```

```
> t(B)
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> A%*%t(B)
     [,1] [,2]
[1,]   22   49
[2,]   28   64
```

BEWARE A*B gives element by element multiplication which is rarely required:–

```
> A*B
     [,1] [,2] [,3]
[1,]    1    6   15
[2,]    8   20   36
```

0.5.1.4 Dimensions & Lengths of Matrices of Vectors

```
> C<-matrix(c(1,2,3,4,5,6),2,3)
> dim(C)
[1] 2 3
> dim(t(C))
[1] 3 2
> length(C)
[1] 6
```

```
> x<-c(1,2,3,4)
> length(x)
[1] 4
> dim(x)
NULL
```

Beware:

```
> x
[1] 1 2 3 4
> t(x)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
> dim(x)
NULL
> dim(t(x))
[1] 1 4
> dim(matrix(t(x)))
[1] 4 1
```

```
> matrix(x)
     [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4
> dim(matrix(x))
[1] 4 1
> t(matrix(x))
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
> dim(t(matrix(x)))
[1] 1 4
```

0.5.1.5 Joining matrices together

```
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> C
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> cbind(A,C)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    1    3    5
[2,]    2    4    6    2    4    6
```

```
> rbind(A,C)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
[3,]    1    3    5
[4,]    2    4    6
```

```
> t(rbind(t(A),t(B)))
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    1    2    3
[2,]    2    4    6    4    5    6
```

```
> t(cbind(t(A),t(B)))
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
[3,]    1    2    3
[4,]    4    5    6
```

Beware:–

Joining non-conformable matrices will generate error messages

0.5.1.6 Diagonals and Trace

```
> E<-matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=T)
> E
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> diag(E)
[1] 1 5 9
> sum(diag(E))
[1] 15
```

```
> diag(c(1,5,9))
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    5    0
[3,]    0    0    9
> diag(diag(E))
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    5    0
[3,]    0    0    9
```

0.5.1.7 Trace of Products

```
> F<-matrix(c(1,2,3,4,5,6,7,8,9),3,3,)
> F
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> F%*%E
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126
```

```
> E%*%F
     [,1] [,2] [,3]
[1,]   14   32   50
[2,]   32   77  122
[3,]   50  122  194
> sum(diag(E%*%F))
[1] 285
> sum(diag(F%*%E))
[1] 285
```

0.5.1.8 Transpose of Products

```
> t(E%*%F)
     [,1] [,2] [,3]
[1,]   14   32   50
[2,]   32   77  122
[3,]   50  122  194
```

```
> t(F)%*%t(E)
     [,1] [,2] [,3]
[1,]   14   32   50
[2,]   32   77  122
[3,]   50  122  194
```

Beware:

```
> t(E)%*%t(F)
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126
```

Note

EF and FE are symmetric but neither E nor F is symmetric. Also $E'F' \neq (EF)'$

0.5.1.9 Determinants

```
> G<-matrix(c(1,-2,2,2,0,1,1,1,-2),3,3,byrow=T)
> G
     [,1] [,2] [,3]
[1,]    1   -2    2
[2,]    2    0    1
[3,]    1    1   -2
> det(G)
[1] -7
```

0.5.1.10 Diagonal matrices

```
> G<-matrix(c(1,-2,2,2,0,1,1,1,-2),3,3,byrow=T)
> G
     [,1] [,2] [,3]
[1,]    1   -2    2
[2,]    2    0    1
[3,]    1    1   -2
> diag(G)
[1]  1  0 -2
```

```
> diag(diag(G))
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    0    0
[3,]    0    0   -2
```

0.5.1.11 Inverses

```
> solve(A%*%t(B))
           [,1]       [,2]
[1,]  1.7777778 -1.3611111
[2,] -0.7777778  0.6111111
```

```
> ginv(A%*%t(B))
           [,1]       [,2]
[1,]  1.7777778 -1.3611111
[2,] -0.7777778  0.6111111
```

Beware:

```
> ginv(t(A)%*%B)
           [,1]        [,2]       [,3]
[1,]  1.7407407  0.46296296 -0.8148148
[2,]  0.2685185  0.07407407 -0.1203704
[3,] -1.2037037 -0.31481481  0.5740741
```

BUT:

```
> solve(t(A)%*%B)
Error in solve.default(t(A) %*% B) : system is computationally singular: reciprocal condition number = 2.03986e-18
```

0.5.1.12 Eigen Analyses

```
> eigen(A%*%t(B))
$values
[1] 85.5793377  0.4206623
$vectors
           [,1]       [,2]
[1,] -0.6104370 -0.9151818
[2,] -0.7920648  0.4030412
```

```
> eigen(t(A)%*%B)
$values
[1] 8.55793e+01 4.20662e-01 9.81191e-16
$vectors
           [,1]       [,2]       [,3]
[1,] -0.2421783 -0.8150419  0.4082483
[2,] -0.5280603 -0.1245978 -0.8164966
[3,] -0.8139422  0.5658463  0.4082483
```

0.5.1.13 Singular Value Decomposition

```
> X<-matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=T)
> X
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> svd(X)
$d
[1] 1.684810e+01 1.068370e+00 1.776048e-16
```

```
$u
           [,1]       [,2]      [,3]
[1,] -0.21483  0.88723  0.40824
[2,] -0.52058  0.24964 -0.81649
[3,] -0.82633 -0.38794  0.40824
```

```
$v
           [,1]       [,2]       [,3]
[1,] -0.47967 -0.776690  0.40824
[2,] -0.57236 -0.075686 -0.81649
[3,] -0.66506  0.625318  0.40824
```

1 Vectors and Matrices

1.1 Vectors

1.1.1 Definitions

A vector x of *order* p (or *dimension*) is a column of p numbers:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}$$

Technically x is an element of p-dimensional Euclidean space $\mathbb{R}^p$ but this will not be central to the material below.

The numbers x_i are the ***components*** (or ***elements***) of x.

It may be referred to as a ***column*** vector for emphasis. The transpose of x, $x'=(x_1, x_2, \ldots, x_p)$ is a ***row*** vector. Vectors are presumed to be column vectors unless specified otherwise.

Addition and subtraction of vectors of the same order is performed element by element

$$x + y = \begin{pmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_p + y_p \end{pmatrix}$$

It is not possible to add or subtract vectors which are not ***conformable***, i.e. which do not have the same dimension or order.

Scalar multiplication of a vector is element b y element:

$$\lambda x = \begin{pmatrix} \lambda x_1 \\ \lambda x_2 \\ \vdots \\ \lambda x_p \end{pmatrix}$$ for any scalar (real number) λ.

Results of addition and subtraction of two vectors result in a vector of the same order. Results of adding to, subtracting from or multiplying a vector by a scalar result in a vector of the same order.

Two vectors x and y are equal if they are of the same order and each pair of corresponding elements are equal, i.e. $x_i = y_i$, for i = 1,2,…,p.

A vector with all elements 0 is denoted by 0, i.e. if $x_i = 0$ for i = 1,2,…,p then x=0.

A vector e_i with i^{th} element 1 and all others 0 is the ***i^{th} unit vector***.

A vector with all elements 1 is denoted by 1_p, i.e. if $x_i = 1$ for i = 1,2,…,p then $x=1_p$. Clearly $\sum_{i=1}^{p} e_i = 1_p$.

1_p is referred to as the ***sum vector*** [because $x_i'1_p = \sum_{i=1}^{p} x_i$, see below].

Note that $x + y = y + x$ (***commutativity***) and $(x + y) + z = x + (y + z)$ (***associativity***).

Vectors cannot be multiplied together in a usual way but a useful scalar function of two vectors is the ***scalar product*** (or ***inner product***). Mathematicians denote this by $\langle x, y \rangle$ and it is defined by $\langle x, y \rangle = \sum_{i=1}^{p} x_i y_i$.

In Statistics it is more usually denoted by $x'y$ (see later under matrix multiplication).

Note that $\langle x, y \rangle$ is a scalar (i.e. an ordinary number).

Example: If x=(1,2,3)′ and y=(4,5,6)′ then $x'y=\langle x, y \rangle=1\times4+2\times5+3\times6=32$

Two vectors are said to ***orthogonal*** if $x'y=\langle x, y \rangle = 0$.

For example if x = (1, 2, 0)′, y = (–6, 3, 0), z = (0, 0, 7) then x, y and z are mutually orthogonal.

Note that $\langle x, x \rangle = x'x = \sum x_i^2$ = sum of the squared elements of x.

1.2 Creating vectors in R

The easiest way to create a vector is to use the function `c(x1, x2,…,xp)`:

```
> a<-c(1,2,3)
> a
[1] 1 2 3
> b<-c(4,5,6)
```

and this 'works' in some cases but a and b will be interpreted as ***either*** a column ***or*** a row vector ***according to context***. It is better to remove ambiguity and ensure that the vector is of the right "class" and force this by making using of the `matrix(.,.,.)` function with 1 column:

```
> c<-matrix(c(3,2,1),3,1,byrow=T)
> d<-matrix(c(6,5,4),nrow=3,ncol=1,byrow=T)
> c
     [,1]
[1,]    3
[2,]    2
[3,]    1
```

```
> d
     [,1]
[1,]    6
[2,]    5
[3,]    4
```

Without the `matrix` function the result is of class "numeric" not "matrix". Note that using the `matrix` function ensures that **R** prints the result as a column vector. An equivalent way of coercing a string of numbers to be of class "matrix" is to use the function `as.matrix(.)`:

```
> b<-c(4,5,6)
> b
[1] 4 5 6
> class(b)
[1] "numeric"
> b<-as.matrix(b)
```

```
> b
     [,1]
[1,]    4
[2,]    5
[3,]    6
> class(b)
[1] "matrix"
```

Note also the use of `nrow` and `ncol` and the default order that `matrix` assumes if they are omitted:

```
> u<-matrix(c(3,2,1),1,3,byrow=T)
> u
     [,1] [,2] [,3]
[1,]    3    2    1
> v<-matrix(c(6,5,4), ncol=1,nrow=3,byrow=T)
```

```
> v
     [,1]
[1,]    6
[2,]    5
[3,]    4
```

When entering column vectors or row vectors the `byrow` argument has no effect, i.e. `byrow=T` and `byrow=F` give the same result but this is not the case when entering matrices (see §1.11).

1.3 Matrices

1.3.1 Definitions

An m × n matrix X is a rectangular array of scalar numbers:

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

This matrix has m rows and n columns, it is an m × n matrix ("m ***by*** n matrix"), it is a matrix of ***order*** m × n, X has dimensions m and n. Technically X is an element of m × n-dimensional Euclidean space $\mathbb{R}^{m\times n}$ but this will not be central to the material below.

Sometimes we may write $X=(x_{ij})$

For example $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ is a 2 × 3 matrix.

A [column] vector is a matrix with one column, it is a n × 1 matrix where n is the order of the vector. A row vector is a 1 × n matrix.

The numbers x_{ij} are the ***components*** (or ***elements***) of X.

A matrix is a ***square matrix*** if m = n, i.e. if it has the same number of rows and columns, i.e. an n × n matrix is a square matrix.

The ***transpose*** of the m × n matrix X is the n × m matrix X′:

$$X' = \begin{pmatrix} x_{11} & x_{21} & \cdots & x_{m1} \\ x_{12} & x_{22} & \cdots & x_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1n} & x_{2n} & \cdots & x_{mn} \end{pmatrix}$$

A square matrix X is ***symmetric*** if X′ = X.

Two matrices X and Y are equal if they are of the same order and each pair of corresponding elements are equal, i.e. $x_{ij} = y_{ij}$, for i = 1,2,…,m and j=1, 2, …,n.

A matrix with all elements 0 is denoted by 0, i.e. if $x_{ij} = 0$ for i = 1,2,…, m and j=1, 2, …,n then X=0.

A square matrix with all elements not on the diagonal equal to 0 is a ***diagonal matrix,*** i.e. if $x_{ij} = 0$ for all $i \neq j$ (and $x_{ii} \neq 0$ for at least one i).

A diagonal matrix with all diagonal elements 1 (and all others not on the diagonal 0) is denoted by I_p, i.e. if $x_{ii} = 1$ for i = 1,2,…,n and $x_{ij} = 0$ for $i \neq j$ for i,j = 1, 2, …, n then $X=I_n$. It is referred to as the ***identity matrix***.

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

Note that X + Y = Y + X (*commutativity*) and (X + Y) + Z = X + (Y + Z) (*associativity*), provided X, Y and Z are all of the same order.

If X is a square matrix then diag(X) is the [column] vector of the diagonal elements of X, i.e. the vector (x_{ii}). If u is a vector then diag(u) is the diagonal matrix with the elements of u along the diagonal and 0s elsewhere. So, diag(diag(X)) is a square matrix formed by setting all off-diagonal elements of X to 0. Some texts will call diag(X) this matrix but the forma diag(diag(X)) here conforms with **R** syntax.

The ***trace*** of a square matrix is the sum of all the diagonal elements of the matrix, i.e. trace(X) = trace(x_{ij}) = $\sum_{i=1}^{n} x_{ij}$.

Note that trace(I_n) = n.

1.4 Matrix Addition, subtraction and scalar multiplication

Addition and subtraction of matrices of the same order is performed element by element (just as with vectors):

$$X + Y = (x_{ij}) + (y_{ij}) = (x_{ij} + y_{ij})$$

It is not possible to add or subtract matrices which do not have the same dimensions or order.

Scalar multiplication of a matrix is element b y element:

$$\lambda X = \lambda(x_{ij}) = (\lambda x_{ij}).$$

Results of addition and subtraction of two matrices result in a vector of the same order. Results of adding to, subtracting from or multiplying a matrix by a scalar result in a matrix of the same order.

e.g. $$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 6 & 8 \\ 10 & 12 \end{pmatrix} = 2\begin{pmatrix} 3 & 4 \\ 5 & 6 \end{pmatrix}$$

1.5 Matrix Multiplication

If A and B are matrices then we can multiply A by B (to get AB) **only if the number of columns of A equals the number of rows of B.** So if A is an $m \times n$ matrix and B is a $n \times p$ matrix then the product AB is can be defined (but not the product BA). The result is a $m \times p$ matrix, i.e. the number of rows of the first matrix and the number of columns of the second. The (i,k)th element of AB is obtained by summing the products of the elements of the ith row of A with the elements of the kth column of B, $AB = \left(\sum_{j=1}^{n} a_{ij}b_{jk}\right)$.

If C is $m\times n$ and D is $p\times q$ then the product CD can only be defined if n=p, in which case C and D are said to be ***conformable.*** If C and D are such that CD is not defined then they are ***non-conformable***.

Examples: if $U = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$, $V = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$ then U is 2×2 and V is 2×2 so UV is $2\times 2\times 2\times 2 \equiv 2\times 2$ and VU is $2\times 2\times 2\times 2 \equiv 2\times 2$

So,

$$UV = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1\times 5+2\times 7 & 1\times 6+2\times 8 \\ 3\times 5+4\times 7 & 3\times 6+4\times 8 \end{pmatrix} = \begin{pmatrix} 5+14 & 6+16 \\ 15+28 & 18+32 \end{pmatrix}$$

$$= \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

and $$VU = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 5+18 & 10+24 \\ 7+24 & 14+32 \end{pmatrix} = \begin{pmatrix} 23 & 34 \\ 31 & 46 \end{pmatrix}$$

Notice that $UV \neq VU$ because $\begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix} \neq \begin{pmatrix} 23 & 34 \\ 31 & 46 \end{pmatrix}$

if $A=\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}$, $B=\begin{pmatrix}1&2\\3&4\\5&6\end{pmatrix}$ then note A is a 2×3 matrix and B is a 3×2 matrix so AB is 2×3×3×2≡2×2 and BA is 3×2×2×3 ≡ 3×3

$$AB=\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}\begin{pmatrix}1&2\\3&4\\5&6\end{pmatrix}=\begin{pmatrix}1\times1+2\times3+3\times5 & 1\times2+2\times4+3\times6\\4\times1+5\times3+6\times5 & 4\times2+5\times4+6\times6\end{pmatrix}$$

$$=\begin{pmatrix}22&28\\49&64\end{pmatrix}$$

$$BA=\begin{pmatrix}1&2\\3&4\\5&6\end{pmatrix}\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}=\begin{pmatrix}1\times1+2\times4 & 1\times2+2\times5 & 1\times3+2\times6\\3\times1+4\times4 & 3\times2+4\times5 & 3\times3+4\times6\\5\times1+6\times4 & 5\times2+6\times5 & 5\times3+6\times6\end{pmatrix}$$

$$=\begin{pmatrix}9&12&15\\19&26&33\\29&40&51\end{pmatrix}$$

Notice that AB ≠ BA and indeed AB is a 2×2 matrix and BA is a 3×3 matrix.

U is 2×2, A is 2×3 and B is 3×2 so U and A are conformable and UA is defined (& is a 2×2×2×3 ≡ 2×3 matrix) but U and B are non-conformable and you cannot calculate UB because it would be 2×**2×3**×2 ≢ anything.

We have $UA = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} = \begin{pmatrix} 9 & 12 & 15 \\ 19 & 26 & 33 \end{pmatrix}$

Note that the transpose of B, B′, is 2×3 so U and B′ are conformable and

we have $UB' = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}' = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 & 3 & 5 \\ 2 & 5 & 6 \end{pmatrix} = \begin{pmatrix} 5 & 11 & 17 \\ 11 & 25 & 39 \end{pmatrix}$.

In the product AB we say that B is ***premultiplied*** by A and A is ***postmultiplied*** by B.

In general for two matrices X and Y we have XY ≠ YX. If we have matrices such that both XY and YX are defined and we XY = YX then we say X and Y ***commute***. If X is m×n and Y is p×q and if both XY and YX are defined then we must have p=n and q=m so XY is n×m×m×n ≡ n×n and YX is m×n×n×m ≡ m×m. Thus if X and Y commute then XY=YX and in particular XY and YX most have the same *orders* so we must m=n and thus two matrices can only commute if they are ***square matrices of the same order***.

Note that square matrices do not in general commute (e.g. U and V above).

If $W = \begin{pmatrix} 2 & 2 \\ 3 & 5 \end{pmatrix}$ then UW = WU and U and W commute (check this for yourself as an exercise). The ***identity matrix*** I_n commutes with all n×n square matrices (check this for yourself) and further all ***diagonal*** matrices commute with matrices of the same order (note that diagonal matrices are necessarily square).

Note that $(A+B)^2 = (A+B)(A+B) = A(A+B)+B(A+B) = A^2+AB+BA+B^2$ and so $(A+B)^2 = =A^2+2AB+B^2$ ***only if*** A and B *commute.*

1.6 Transpose and Trace of Sums and Products.

If A+B is defined (i.e. A & B have the same orders) then (A+B)′ = A′+B′ and trace(A+B) = trace(A)+trace(B).

If A is m×n and B is n×p then A×B is m×n×n×p = m×p so (AB)′ is p×m.

It is easy to show **(AB)′ = B′A′** :

$$(AB)' = ((a_{ij})(b_{jk}))' = (\sum_{j=1}^{n} a_{ij}b_{jk})' = (\sum_{j=1}^{n} a_{kj}b_{ji}) = (b_{jk})'(a_{ij})' = B'A'$$

Note that A′ is n×m and B′ is p×n so the product A′B′ is not defined but B′A′ is defined.

If λ s any scalar then (λA)′ = λA′.

If A is m×n and B is n×m (so both AB and BA are defined, i.e. A and B are mutually conformable) then **trace(AB) = trace(BA)**:

We have $AB = (\sum_{j=1}^{n} a_{ij}b_{jk})$ so $\text{trace}(AB) = (\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}b_{ji})$

$BA = (\sum_{i=1}^{m} b_{ji}a_{ik})$ so $\text{trace}(BA) = (\sum_{j=1}^{n}\sum_{i=1}^{m} b_{ji}a_{ij})$=trace(AB).

If λ is any scalar then trace(λA) = λtrace(A).

1.7 Special Matrices

1.7.1 Symmetric & Skew-symmetric matrices

A square n×n matrix A = (a_{ij}) is *symmetric* of A′ = A , i.e. $a_{ij} = a_{ji}$ for all i, j.

A square matrix B is ***skew-symmetric*** if B′ = –B, i.e. $b_{ij} = -b_{ji}$, all i, j. It is easy to see that all skew-symmetric matrices have zero elements on the diagonals.

Any square matrix X can be expressed a X = ½(X+X′)+½(X–X′). Since (X+X′)′ = (X+X′) and (X–X′)′ = –(X–X′) we have that any square matrix can be expressed as the sum of a *symmetric part* and a *skew-symmetric part*.

Let $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ then both A and B are symmetric but

$AB = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$ which is **not** symmetric.

Consider Z = A′BA. If B is symmetric then Z is symmetric, because (A′BA)′ = A′B′**(**A′)′ **=** A′BA if B is symmetric and B′ =B. However, the converse is not true:

Let $A = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$, then A′ = $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and

A′BA = $\begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ b_{11} & b_{12} \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & b_{11} \end{pmatrix}$ whether or not B is symmetric.

1.7.2 Products with Transpose AA′

If A is $m\times n$ then the products AA′ and A′A are both defined, resulting in $m\times m$ and $n\times n$ matrices respectively. Both are symmetric, because, for example, $(AA')' = (A')'A' = AA'$.

If the columns of A are $a_{.j}$ then $AA' = \Sigma_j a_{.j} a_{.j}'$.

(For a general product AB we have $AB = \Sigma_j a_{.j} b_{j.}$ where the $b_{j.}$ are the rows of B.)

1.7.3 Orthogonal Matrices

A $p\times p$ square matrix A is orthogonal if $A'A = AA' = I_p$.

Note that for square matrices if $A'A = I_p$ then necessarily we have $AA' = I_p$ since if $A'A = I_p$ then $(A')^{-1}A'AA' = (A')^{-1}I_pA' = I_p$ (see later for section on inverses).

Also $A^{-1} = A'$ and if A is orthogonal then A′ is also orthogonal.

If A and B are both orthogonal and both $p\times p$ then AB is orthogonal because $(AB)'AB = B'A'AB = B'I_pB = B'B = I_p$.

[It is possible to have a $m\times n$ matrix B such that $B'B = I_n$ but $BB' \neq I_m$, e.g. the 2×1 matrix $B = (1,0)'$ has $B'B = 1$ and $BB' \neq I_2$].

1.7.3.1 Examples:

$A = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$, $B = \begin{pmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{pmatrix}$ are both orthogonal

$A = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$, $B = \begin{pmatrix} \cos\theta & -\sin\theta \\ -\sin\theta & -\cos\theta \end{pmatrix}$ are both orthogonal for any value of θ and it can be shewn that any orthogonal 2×2 matrix is of this form.

$A = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 & 1 \\ 0 & 1 & 2 \\ 1 & 1 & -1 \end{pmatrix}$ is orthogonal.

If A is a $p\times p$ matrix with rows $a_{1\cdot}, a_{2\cdot}, \ldots, a_{p\cdot}$ and columns $a_{\cdot 1}, a_{\cdot 2}, \ldots, a_{\cdot p}$ then $a_{i\cdot}'a_{j\cdot} = 1$ if $i = j$ and 0 if $i \neq j$ and $a_{\cdot i}a_{\cdot j}' = 1$ if $i = j$ and 0 if $i \neq j$, i.e. the rows and columns of A are orthogonal.

1.7.3.2 Normal Matrices

A $p\times p$ matrix is ***normal*** if $A'A = AA'$, i.e. if A commutes with A′**.**

Clearly all symmetric, all skew-symmetric and all orthogonal matrices are normal.

1.7.4★★ Permutation Matrices

The n×n matrix A is a ***permutation*** matrix if each row and each column has exactly one 1 and the rest of the entries are zero.

Examples: $A_1=\begin{pmatrix}1&0\\0&1\end{pmatrix},\ A_2=\begin{pmatrix}0&1\\1&0\end{pmatrix}, B_1=\begin{pmatrix}0&0&1\\0&1&0\\1&0&0\end{pmatrix}, B_2=\begin{pmatrix}0&1&0\\1&0&0\\0&0&1\end{pmatrix}, I_3$

The columns of a permutation matrix are the complete set of unite vectors e_i taken in some order, i.e. not necessarily $e_1, e_2,\dots, e_p$. For example $B_1=(e_3, e_2, e_1)$, $B_2=(e_2, e_1, e_3)$.

The effect of pre-multiplication of a matrix X by a permutation matrix is to permute the rows of X, post-multiplication permutes the columns.

All permutation matrices are orthogonal.

1.7.5+ Idempotent Matrices

A p×p matrix A is ***idempotent*** if $A^2 = I_p$. Clearly I_p and $0_{p\times p}$ are idempotent and if p=2 then these are the only 2×2 idempotent matrices.

Let $H_n=I_n-\tfrac{1}{n}1_n1_n'$ then $H_n^2=(I_n-\tfrac{1}{n}1_n1_n')=I_n-\tfrac{1}{n}1_n1_n'-\tfrac{1}{n}1_n1_n'(I_n-\tfrac{1}{n}1_n1_n')$

$=I_n-\tfrac{1}{n}1_n1'-\tfrac{1}{n}1_n1_n'I_n+(\tfrac{1}{n}1_n1_n')(\tfrac{1}{n}1_n1_n')=H_n-\tfrac{1}{n}1_n1'+\tfrac{1}{n^2}1_n(1_n'1_n)1_n'$

and $(1_n'1_n)$ is 1×n×n×1 ≡ 1×1, i.e. a scalar and = n, noting that 1_n is the *sum vector* which sums the elements of a vector when post-multiplying it (or pre-multiplying it by its transpose). So $H_n^2=H_n-\tfrac{1}{n}1_n1'+\tfrac{1}{n}1_n1_n'=H_n$ and H_n is idempotent. H_n is called the ***centering matrix.***

1.8⁺ Partitioned Matrices

1.8.1⁺ Sub-matrices

A ***sub-matrix*** of a m×n matrix A is a rectangular $m_1 \times n_1$ section of it forming a $m_1 \times n_1$ matrix A_{11}. Note that some texts will regard any matrix formed by a subset of rows and columns (not necessarily contiguous) as a sub-matrix of A. A sub-matrix A_{ij} of A can be expressed as a product EAF where E and F are matrices with all entries either 0 or 1. For example, if suppose A_{11} is the top left hand corner of A consisting of m_1 rows and n_1 columns. Let E be the $m_1 \times n$ matrix with a 1 in cells (1,1), (2,2), …, (m_1,m_1) and 0 elsewhere and let F be the $m \times n_1$ matrix with 1 in cells (1,1), (2,2), …, (n_1,n_1) and 0 elsewhere. Then EAF = A_{11}.

1.8.1.1 Example

```
A<- matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=T)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> E<-matrix(c(1,0,0,0,1,0),2,3,byrow=T)
> E
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
> E%*%A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> F<- matrix(c(1,0,0,1,0,0),3,2,byrow=T)
> F
     [,1] [,2]
[1,]    1    0
[2,]    0    1
[3,]    0    0
> A%*%F
     [,1] [,2]
[1,]    1    2
[2,]    4    5
[3,]    7    8
> E%*%A%*%F
     [,1] [,2]
[1,]    1    2
[2,]    4    5
```

1.8.2+ Manipulation of partitioned matrices

A matrix A could be partitioned in four sub-matrices A_{11}, A_{12}, A_{21} and A_{22} with orders $m_1 \times n_1$, $m_1 \times n_2$, $m_2 \times n_1$ and $m_2 \times n_2$, with $m_1+m_2=m$ and $n_1+n_2=n$, so we would have

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

This can be useful if some of the A_{ij} have a special form, e.g. are 0 or diagonal or the identity matrix. Adding and multiplying matrices expressed in their partitioned form is performed by rules analogous to those when adding and multiplying element by element.

For example, if B is partitioned into B_{11}, B_{12}, B_{21} and B_{22} with orders $m_1 \times n_1$, $m_1 \times n_2$, $m_2 \times n_1$ and $m_2 \times n_2$, and C is partitioned in C_{11}, C_{12}, C_{21} and C_{22} with orders $n_1 \times p_1$, $n_1 \times p_2$, $n_2 \times p_1$ and $n_2 \times p_2$ (so A_{ij} and B_{ij} are *addition conformable* and A_{ij} and C_{ij} are *multiplication conformable*) then we have

$$A + B = \begin{pmatrix} A_{11} + B_{11} & A_{12} + B_{12} \\ A_{21} + B_{21} & A_{22} + B_{22} \end{pmatrix}, \quad AC = \begin{pmatrix} A_{11}C_{11} + A_{12}C_{21} & A_{12}C_{12} + A_{12}C_{22} \\ A_{21}C_{11} + A_{22}C_{21} & A_{21}C_{21} + A_{22}C_{22} \end{pmatrix}$$

which symbolically are identical to the form we would have if $m_1=m_2=n_1=n_2=p_1=p_2=1$.

Further we have $A' = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}' = \begin{pmatrix} A'_{11} & A'_{21} \\ A'_{12} & A'_{22} \end{pmatrix}$ (*note the interchange of the two off-diagonal blocks*).

1.8.2.1+ Example

A matrix can be partitioned into its individual columns:

Let X=(x_1, x_2, …, x_n) where x_i are p-vectors, i.e. X is a p×n matrix. Then

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix} \text{ and } X' = \begin{pmatrix} x'_1 \\ x'_2 \\ \vdots \\ x'_n \end{pmatrix},$$

and is a n×p matrix 'partitioned' into n rows.

Further we have

$XX' = \sum_{k=1}^{n} x_k x'_k$ and $X'X = (x'_i x_j)$ (Note that XX′ is a p×p matrix whose (i,j)th element is $\sum_{k=1}^{n} x_{ki} x_{kj}$ and X′X is a n×n matrix whose (i,j)th element is $x'_i x_j$, the inner product of x_i and x_j , i.e. $\sum_{k=1}^{p} x_{ik} x_{jk}$.

A partitioned matrix of the form $\begin{pmatrix} Z_{11} & 0 & 0 & 0 \\ 0 & Z_{22} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & Z_{rr} \end{pmatrix}$ where Z_{ii} is a $m_i \times n_i$ matrix and the zero sub-matrices are of conformable orders is called a ***block diagonal matrix***.

1.8.3+ Implementation of partitioned matrices in R

Matrices of conformable dimensions can be joined together horizontally and vertically by command `cbind(.,.)` and `rbind(.,.)`. A sub-matrix of a m×n matrix A of size $m_1 \times n_1$ [in the top left corner] can be specified by `A[1:m1,1:n1]`.

```
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> U<-cbind(A,B)
> U
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    1    2    3
[2,]    2    4    6    4    5    6

> U[1:2,3:6]
     [,1] [,2] [,3] [,4]
[1,]    5    1    2    3
[2,]    6    4    5    6

> U[,3:6]
     [,1] [,2] [,3] [,4]
[1,]    5    1    2    3
[2,]    6    4    5    6
```

```
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> V<- rbind(A,B)
> V
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
[3,]    1    2    3
[4,]    4    5    6
> V[1:2,1:3]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> V[2:4,]
     [,1] [,2] [,3]
[1,]    2    4    6
[2,]    1    2    3
[3,]    4    5    6
```

Note that because U has 2 rows `U[1:2,3:6]` and `U[,3:6]` refer to the same sub-matrices (as do `V[2:4,]` and `V[2:4,1:3]` since V has 3 columns.

1.9 Algebraic Manipulation of Matrices

1.9.1 General Expansion of Products

An expression such as (A+B)(X+Y) can be multiplied out term by term but remember to preserve the order of the multiplication of the matrices. So (A+B)(X+Y)=A(X+Y)+B(X+Y)=AX+AY+BX+BY and this cannot in general be simplified further.

In particular, $(A+B)^2=(A+B)(A+B)=A(A+B)+B(A+B)=A^2+AB+BA+B^2$ and unless A and B commute we can go no further in collating terms.

1.9.2 Useful Tricks

1.9.2.1 Tracking Dimensions & 1×1 Matrices

It can happen that in an expression involving the product of several matrices and vectors some element or sub-product is a 1×1 matrix (i.e. a scalar). This permits two operations of just this part of the product:

(i) it can commute with other terms

(ii) it can be replaced by its transpose

For example suppose S is p×p and x and y are both p-vectors (i.e. p×1). Let A = Sxx′ then A is p×p×p×1×1×p ≡ p×p. Let B = Sx then B is p×p×p×1≡ p×1.

AB = (Sxx′)Sx = Sx(x′Sx) which is (p×p×p×1)×(1×p×p×p×p×1). The second factor x′Sx is (1×p×p×p×p×1) ≡ 1×1, i.e. a scalar and so commutes with the first factor. So (Sxx′)Sx = (x′Sx)Sx.

This is of the form of matrix×vector = scalar×vector (the same vector) and is referred to as an *eigenequation* (see a later section).

Another example is used in the next section: consider x′Ax which is 1×1, i.e. a scalar and so is symmetric, so x′Ax = (x′Ax)′ = x′A′x, so we have x′Ax = ½(x′Ax+x′A′x) = ½x′(A+A′)x and the matrix ½(A+A′) is symmetric.

1.9.2.2 Trace of Products

Recall that trace(AB)= trace(BA) (if both products are defined) (see§1.6). This is useful either if one of AB and BA is simple (e.g. a scalar) or some other simplification is possible. For example if x is a p-vector then trace(xx′) = trace(x′x) and xx′ is p×p but x′x is 1×1, i.e. a scalar and so possibly easier to evaluate.

An example in the other direction is evaluating y′Sy by noting that y′Sy = trace(y′Sy) = trace(yy′S) = trace(Syy′). A trick like this is used in working with the maximized likelihood of the multivariate normal distribution.

1.10 Linear and Quadratic Forms

If a and x are p-vectors then the inner product $a'x = a_1x_1+a_2x_2+\ldots+a_px_p$ is termed a ***linear form in* x**; it is presumed that a is a known vector of numbers and x is a variable. If A is a p×p matrix then x′Ax is a ***quadratic form in* x**. Again it is presumed that A is a known matrix of numbers and x is a variable. Note that x′Ax is 1×1, i.e. a scalar and so is symmetric, so x′Ax = (x′Ax)′ = x′A′x, so we have x′Ax = ½(x′Ax+x′A′x) = ½x′(A+A′)x and the matrix ½(A+A′) is symmetric. So, we need only consider the properties of quadratic forms which involve a symmetric matrix.

If x′Ax > 0 whatever the value of x then A is said to be ***positive definite*** and if x′Ax ≥ 0 for all x then A is said to be ***positive semi-definite,*** similarly ***negative definite*** and ***negative semi-definite*** if x′Ax<0 or ≤0.

It is always assumed that if these terms are used then A is symmetric.

See later for more on quadratic forms.

Example: Suppose p=3, then $x'Ax =$

$$a_{11}x_1^2 + a_{22}x_2^2 + a_{33}x_3^2 + a_{12}x_1x_2 + a_{21}x_1x_2 + a_{13}x_1x_3 + a_{31}x_1x_3 + a_{23}x_2x_3 + a_{32}x_2x_3$$
$$= a_{11}x_1^2 + a_{22}x_2^2 + a_{33}x_3^2 + (a_{12} + a_{21})x_1x_2 + (a_{13} + a_{31})x_1x_3 + (a_{23} + a_{32})x_2x_3$$

= ½x′(A+A′)x.

If $A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 5 & 9 \end{pmatrix}$ then x′Ax = $x_1^2 + 5x_2^2 + 9x_3^2 + 6x_1x_2 + 10x_1x_3 + 13x_2x_3$

$= x_1^2 + 5x_2^2 + 9x_3^2 + 2\times 3x_1x_2 + 2\times 5x_1x_3 + 2\times 6.5x_2x_3$ so we have

$x'\begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 5 & 9 \end{pmatrix}x = x'\begin{pmatrix} 1 & 3 & 5 \\ 3 & 5 & 6.5 \\ 5 & 6.5 & 9 \end{pmatrix}x$, replacing the asymmetric matrix A with a symmetric one.

1.11 Creating Matrices in R

Small matrices can be entered directly using the function `matrix(.)`. Large matrices may best be entered by reading the data from a file into a `dataframe` (e.g. with `read.table(.)` or `scan(.)`) and then converting to a matrix.

`A<- matrix(c(x11,…,xm1, x12,…,xm2,…x1n,…,xmn), nrow=m, ncol=n, byrow=F)`

creates a m×n matrix taking the values column by column.

`B<- matrix(c(x11,…,x1n, x21,…,x2n,…xm1,…,xmn), nrow=m, ncol=n, byrow=T)`

creates a m×n matrix taking the values row by row. If the `byrow` is omitted then `byrow=F` is assumed. If the terms `nrow=` and `ncol=` are omitted (and just the values `m` and `n` given) then it is assumed they are in the order row and column).

So

`A<- matrix(c(`$x_{11}, \ldots, x_{m1}, x_{12}, \ldots, x_{m2}, \ldots x_{1n}, \ldots, x_{mn}$`), m, n)`

creates a m×n matrix taking the values column by column. See the examples in the next section.

If data are read into a *dataframe* then this can be converted to a matrix by `data.matrix(.)` or `as.matrix(.)`:

```
X<-read.table(filename)
X<-data.matrix(X)
```

Initially `X` is of class "dataframe" and then this is converted to class "matrix", `as.matrix(X)`will have the same effect but can also be used on objects which are not entirely numeric.

The reliance of the class of an object is that some commands will accept arguments only of certain classes.

There are many other ways of creating matrices, some other functions give a matrix as a result, e.g. `eigen(.)` produces a matrix of eigenvectors, `cbind(.,.,.)` will join vectors together into a matrix. Details of these are not given here and the `help` system is generally most informative.

1.12 Examples

1.12.1 Entering matrices

```
> A<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=F)
> B<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Note that the columns of A were filled successively (because `byrow=F`) and the rows of B were filled successively (because `byrow=T`).

```
> C<-matrix(c(1,2,3,4,5,6),2,3)
> C
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Note that the columns of C were filled successively (because `byrow=F` was assumed by default).

```
> D<-matrix(c(1,2,3,4,5,6),2,3,byrow=T)
> D
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

The rows of D were filled successively (because `byrow=T` was specified. In both cases 2 rows and 3 columns were taken since the order rows, columns was assumed.

The order can be overridden by specifying the parameters:

```
> E<-matrix(c(1,2,3,4,5,6),ncol=2,nrow=3,byrow=T)
> E
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

If the parameters are omitted entirely then `ncol=1` is assumed and a column vector is created (which has class `"matrix"` because of the `matrix` function).

```
>  F<-matrix(c(1,2,3,4,5,6))
> F
     [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4
[5,]    5
[6,]    6
```

1.12.1.1 Ambiguity of Vectors

Note that entering vectors without specifying whether they are column or row vectors can lead to ambiguities: if we create the vector a by

```
> a<-c(1,2,3)
```

then it may be interpreted as a column vector (i.e. a 3×1 matrix) or as a row vector (i.e. a 1×3 matrix) according to what operation is being attempted: **R** will do its very best to produce some answer and avoid giving an error message. For example if X is a 3×3 matrix then premultiplying X by a will cause a to be assumed to be a row vector but postmultiplying X by a will cause **R** to regard a as a column vector. See §1.12.3.5.

1.12.2 Addition, Subtraction and Transpose

```
> A+C
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    4    8   12
```

```
> A-D
     [,1] [,2] [,3]
[1,]    0    1    2
[2,]   -2   -1    0
```

Beware:

```
> A+E
Error in A + E : non-conformable arrays
```

```
> E
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

```
>t(E)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

So A and E′ are conformable

```
> A+t(E)
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    4    8   12
```

1.12.3 Multiplication

1.12.3.1 Standard multiplication

```
> A<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
> B<-matrix(c(1,2,3,4,5,6),nrow=3,ncol=2,byrow=T)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6

> B
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6

> A%*%B
     [,1] [,2]
[1,]   22   28
[2,]   49   64

> B%*%A
     [,1] [,2] [,3]
[1,]    9   12   15
[2,]   19   26   33
[3,]   29   40   51
```

1.12.3.2 Element by element multiplication

BEWARE If A and B have the same numbers of rows and columns then A*B gives element by element multiplication which is rarely required:–

```
> A*B
     [,1] [,2] [,3]
[1,]    1    6   15
[2,]    8   20   36
```

1.12.3.3 Non-commuting matrices

```
> U<-matrix(c(1,2,3,4),2,2,byrow=T)
> V<-matrix(c(5,6,7,8),2,2,byrow=T)
> U
     [,1] [,2]
[1,]    1    2
[2,]    3    4

> V
     [,1] [,2]
[1,]    5    6
[2,]    7    8

> U%*%V
     [,1] [,2]
[1,]   19   22
[2,]   43   50

> V%*%U
     [,1] [,2]
[1,]   23   34
[2,]   31   46
```

1.12.3.4 Commuting matrices

```
> W<-matrix(c(2,2,3,5),2,2,byrow=T)
> W
     [,1] [,2]
[1,]    2    2
[2,]    3    5
```

```
> U%*%W
     [,1] [,2]
[1,]    8   12
[2,]   18   26
```

```
>  W%*%U
     [,1] [,2]
[1,]    8   12
[2,]   18   26
```

Beware:

```
> U%*%A
     [,1] [,2] [,3]
[1,]    9   12   15
[2,]   19   26   33
```

But

```
> B
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

```
>
> U%*%B
Error in U %*% B : non-conformable arguments
> t(B)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> U%*%t(B)
     [,1] [,2] [,3]
[1,]    5   11   17
[2,]   11   25   39
```

1.12.3.5 Transpose of products

```
> t(U%*%V)
     [,1] [,2]
[1,]   19   43
[2,]   22   50
```

```
> t(V)%*%t(U)
     [,1] [,2]
[1,]   19   43
[2,]   22   50
```

```
> t(U)%*%t(V)
     [,1] [,2]
[1,]   23   31
[2,]   34   46
```

So (UV)′=V′U′ ≠ U′V′

```
> t(U%*%W)
     [,1] [,2]
[1,]    8   18
[2,]   12   26
```

```
> t(W)%*%t(U)
     [,1] [,2]
[1,]    8   18
[2,]   12   26
```

```
> t(U)%*%t(W)
     [,1] [,2]
[1,]    8   18
[2,]   12   26
```

Note that U and W commute so it follows that U′ and W′ also commute.′

```
> t(W)%*%U
     [,1] [,2]
[1,]    6   13
[2,]   14   29
```

```
> t(W)%*%U
     [,1] [,2]
[1,]   11   16
[2,]   17   24
```

But it does **not** follow that because U and W commute then W′ also commutes with U, as the example above demonstrates.

1.12.3.6 Ambiguity of vectors

Consider the following:

```
> X<-matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=T)
> X
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> a<-c(1,2,3)
> a%*%X
     [,1] [,2] [,3]
[1,]   30   36   42
```

```
> b<-c(4,5,6)
> X%*%b
     [,1]
[1,]   32
[2,]   77
[3,]  122
```

So a is interpreted as a row vector but b as column vector above.

```
> a%*%b
     [,1]
[1,]   32
```

```
> b%*%a
     [,1]
[1,]   32
```

and here the inner product of a and b is returned whatever the order of the product .

To force a and b to column and row vectors see

```
> a<-matrix(a,3,1)
> a
     [,1]
[1,]    1
[2,]    2
[3,]    3
> b%*%a
     [,1]
[1,]   32
```

```
> b<-matrix(b,1,3)
> b
     [,1] [,2] [,3]
[1,]    4    5    6

> a%*%b
     [,1] [,2] [,3]
[1,]    4    5    6
[2,]    8   10   12
[3,]   12   15   18
```

1.12.4 Diagonal Matrices

1.12.4.1 Creating a Diagonal Matrix from a list

```
> F<-diag(c(1,2,3,4,5))
> F
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    2    0    0    0
[3,]    0    0    3    0    0
[4,]    0    0    0    4    0
[5,]    0    0    0    0    5
```

1.12.4.2 Extracting the Diagonal

```
> E%*%A
     [,1] [,2] [,3]
[1,]    9   12   15
[2,]   19   26   33
[3,]   29   40   51
> diag(E%*%A)
[1]  9 26 51
```

1.12.4.3 Converting a Matrix to Diagonal

```
> diag(diag(E%*%A))
     [,1] [,2] [,3]
[1,]    9    0    0
[2,]    0   26    0
[3,]    0    0   51
```

1.12.5 Trace

1.12.5.1 Trace of Square Matrix

```
> sum(diag(E%*%A))
[1] 86
> sum(diag(U))
[1] 5
> V
     [,1] [,2]
[1,]    5    6
[2,]    7    8
> sum(diag(V))
[1] 13
```

But be careful, sum(V) gives the sum of ***all*** elements in the matrix, not just the diagonals.

```
> sum(V)
[1] 26
```

1.12.5.2 Trace of Transpose and Products

```
> sum(diag(U)); sum(diag(t(U)))
[1] 5
[1] 5
>
> U%*%V
     [,1] [,2]
[1,]   19   22
[2,]   43   50
```

```
> V%*%U
     [,1] [,2]
[1,]   23   34
[2,]   31   46
```

But

```
> sum(diag(U%*%V)); sum(diag(V%*%U))
[1] 69
[1] 69
```

1.12.5.3 Creating a Function for Trace of a Matrix

```
> tr<-function(X) { sum(diag(X)) }
> tr(U)
[1] 5
> tr(t(U))
[1] 5
> tr(U%*%V);tr(V%*%U)
[1] 69
[1] 69
```

1.13 Exercises 1

1. Let $a=\begin{pmatrix}1\\2\\3\end{pmatrix}$, $b=\begin{pmatrix}4\\5\\6\end{pmatrix}$, $u=\begin{pmatrix}3\\2\\1\end{pmatrix}$, $v=\begin{pmatrix}6\\5\\4\end{pmatrix}$, $w=(7,8,9)$
   a) calculate a + b, v – a, w′+ b, 3u, w – a′, v/3, ab′ and ba′
   b) repeat the calculations in a) using R

2. Let $x=\begin{pmatrix}2\\2\\-3\end{pmatrix}$ and $y=\begin{pmatrix}1\\-2\\1\end{pmatrix}$
   a) Which of a, b, u, v, w in Q1 are orthogonal to x?
   b) Which of a, b, u, v, w in Q1 are orthogonal to y?
   c) Check the answers to a) and b) using R

3. Let $A=\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}$, $B=\begin{pmatrix}1&2\\3&4\\5&6\end{pmatrix}$, $U=\begin{pmatrix}1&2\\3&4\end{pmatrix}$, $V=\begin{pmatrix}5&6\\7&8\end{pmatrix}$,

   $W=\begin{pmatrix}2&2\\3&5\end{pmatrix}$ and $Z=\begin{pmatrix}3&2\\3&6\end{pmatrix}$ (& use the vectors from Q1 and Q2)
   a) Find AB, B′A′, BA, a′A, a′Aa, Vdiag(U), diag(B′A′), UVWZ, diag(diag(UV)), diag(diag(U))diag(diag(V)).
   b) Verify U and V do not commute, but U and W commute and U and Z commute. Do W and Z commute? (Guess & verify).

4. Using the matrices from Q3,
   a) calculate x′Ux, x′Vx, x′BAx and x′A′B′x
   b) write the four results in the form x′Sx where S is ***symmetric.***

5. Let $A=\begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}$, $B=\begin{pmatrix}0 & 1\\ 0 & 0\end{pmatrix}$, $C=\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$, $D=\begin{pmatrix}1 & -1\\ -1 & -1\end{pmatrix}$,
   $E=\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}$ and $F=\begin{pmatrix}1 & 1\\ -1 & -1\end{pmatrix}$ then show:–
   a) $A^2 = -I_2$ (so A is 'like' the square root of –1)
   b) $B^2 = 0$ (but $B \neq 0$)
   c) $CD = -DC$ (but $CD \neq 0$)
   d) $EF = 0$ (but $E \neq 0$ and $F \neq 0$)

1.13.1★★ Notes

Q5 illustrates that some rules of scalar multiplication do not carry over to matrix multiplication. However there are some analogies:

(a) If a real square matrix A is such that A′A = 0 then we must have A = 0 because the (i, j)th element of A′A is $\sum_{k=1}^{n} a_{ki}a_{kj}$ so the diagonal element is $\sum_{k=1}^{n} a_{kj}^2$ so if A′A = 0 then in particular the diagonal elements of A′A are zero so we have $\sum_{k=1}^{n} a_{kj}^2 = 0$ and thus $a_{kj} = 0$ for all k and j and so A = 0.

(b) AB = 0 if and only if A′AB = 0 since if A′AB = 0 then B′A′AB = 0 so (AB)′(AB) = 0 and the result follows by (a).

(c) AB = AC if and only if A′AB = A′AC which follows by replacing B by B – C.

2 Rank of Matrices

2.1 Introduction and Definitions

An $m \times n$ matrix X has n columns, x_1, x_2, …, x_n, each of which are [column] vectors of length m (or more technically they are elements of $\mathbb{R}^m$) and it has m rows, each of which are [row] vectors of length n.

Two vectors x_1 and x_2 are *linearly independent* if $a_1x_1 + a_2x_2 = 0$ (where a_1 and a_2 are real numbers) implies $a_1 = a_2 = 0$.
A set of vectors x_1, x_2, …, x_r is *linearly independent* if $\Sigma a_ix_i = 0$ implies all $a_i = 0$ or, in words, they are linearly independent if *there are no non-trivial linear combinations of them which equal zero.*

The [**column-**]**rank of X** is the maximum number of linearly independent columns of X. The [row-]rank of X is the maximum number of linearly independent rows of X. The row-rank of X is clearly the same as the column-rank of X′ (the transpose of X).

A key theorem, which is non-trivial to prove, is that the row rank and the column rank of a matrix are equal. Thus we can talk unambiguously about the ***rank of a matrix*** **X** (written **rk(X)**) without specifying whether we mean row-rank or column-rank. (The most straightforward proof relies on the notions of a dimension of a vector space and is beyond the immediate needs of this introductory material for statistical multivariate analysis).

Clearly the [column-] rank of X ≤ n and also the [row-] rank of X ≤ m so we have **rk(X)** ≤ **min(m,n).**

2.2 Examples

(i) Let $X = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$ then X is a 2×3 matrix so rk(X) ≤ min(2,3) = 2.

So rk(X) = either 1 or 2. If rk(X) = 1 then the rows of X are *linearly independent*, i.e. there are constants a_1 and a_2 (≠ 0) such that a_1 (1, 3, 5) + a_2 (2, 4, 6) = 0. Thus we need $a_1 + 2a_2 = 0$, $3a_1 + 4a_2 = 0$ and $5a_1 + 6a_2 = 0$. Subtracting 3 times the first equation from the second yields $-2a_2 = 0$ so we have $a_1 = a_2 = 0$ so the rows of X are linearly independent and rk(X) ≥ 2, thus rk(X) = 2.

(ii) Let $X = \begin{pmatrix} 4 & 6 \\ 6 & 9 \end{pmatrix}$. X is 2×2 so rk(X) ≤ 2. If a_1 (4, 6) + a_2 (6, 9) = 0 then we have $4a_1 + 6a_2 = 0$ (i.e. $2a_1 + 3a_2 = 0$) and $6a_1 + 9a_2 = 0$, i.e. ($2a_1 + 3a_2 = 0$ again) so we can take $a_1 = 3$ and $a_2 = -2$ and so the columns of X are linearly dependent and thus rk(X) < 2 but rk(X) ≥ 1 so we conclude rk(X) = 1.

(iii) Let $X = \begin{pmatrix} 1 & 2 & 3 & 2 \\ 4 & 5 & 6 & -1 \\ 5 & 7 & 9 & 1 \end{pmatrix}$ then X is 3×4 so rk(X) ≤ min(3,4) = 3.

Looking at X it is easy to see that the first row plus the second row is equal to the third row so the rows are not linearly independent, thus rk(X) < 3. If rk(X) = 1 then each row must be a multiple (possibly fractional) of every other row and again it is easy to see that this is not so and thus rk(X) ≥ 2 and we conclude rk(X) = 2.

(iv) Let $X = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 1 & 5 \\ 6 & 4 & 5 \\ 3 & 1 & 4 \end{pmatrix}$ then X is 4×3 so rk(X) ≤ min(4,3) = 3.

It is easy to see that the first two columns of X are linearly independent (otherwise one would be a multiple of the other) (so certainly rk(X) ≥ 2) but not so easy to tell whether all three columns are linearly independent. Suppose $X(a_1,a_2,a_3)' = 0$ then we have $a_1 + 2a_2 + 3a_3 = 0$, $5a_1 + a_2 + 5a_3 = 0$, $6a_1 + 4a_2 + 5a_3 = 0$ and $3a_1 + a_2 + 4a_3 = 0$. Subtracting multiples of the first from the second and third gives $-9a_2 - 10a_3 = 0$ and $-8a_2 - 13a_3 = 0$. Eliminating a_2 from these shews $a_3 = 0$ and hence $a_2 = a_3 = 0$ and so the columns of X are linearly independent and thus rk(X) = 3.

(v) Let $X = \begin{pmatrix} 1 & 5 & 6 \\ 2 & 6 & 8 \\ 7 & 1 & 8 \end{pmatrix}$ then rk(X) ≤ 3 necessarily and it is easy to see that the first two columns of X are linearly independent and so rk(X) ≥ 2.

Suppose $X(a_1,a_2,a_3)' = 0$ then we have $a_1 + 5a_2 + 6a_3 = 0$, $2a_1 + 6a_2 + 8a_3 = 0$ and $7a_1 + a_2 + 8a_3 = 0$. Using the first equation to eliminate a_1 from the second and third gives $-4a_2 - 4a_3 = 0$ and $-34a_2 - 34a_3 = 0$ and so we can take $(a_1,a_2,a_3)' = (1,1,-1)'$ to satisfy $X(a_1,a_2,a_3)' = 0$ non-trivially shewing rk(X) < 3, thus rk(X) = 2.

2.2.1 Notes

An $m{\times}n$ matrix X with rk(X) = min(m,n) is said to be of ***full rank*** (sometimes *full row-rank or full column-rank* as appropriate).

It is clear that it is not always easy to determine the rank of a matrix, nor even whether it is of full rank, using elementary definitions as above. In practice the easiest methods are to use results which come later in these notes. In particular to determine whether a square $n{\times}n$ matrix is of full rank, one can evaluate its determinant (using `det(X)` in R) and the result that the determinant is non-zero if and only if X is of full rank, see §3.2.2(v). To find the exact rank of any symmetric square matrix the result that the rank is equal to the number of non-zero eigenvalues (see §5.4.7) is useful and this can be easily checked in **R** with the function `eigen(X)`. For general $m{\times}n$ matrices the rank can be checked as the number of non-zero singular values of X given by `svd(X)`, see §5.5.2, or as the number of non-zero eigenvalues either XX' or $X'X$.

2.3 More Examples

(i) If I_n is the $n{\times}n$ identity matrix (i.e. $n{\times}n$, diagonal elements all 1 and all off-diagonal elements 0) then $rk(I_n) = n$ since $I_na = a$ for all vectors a so $I_na = 0$ implies $a = 0$ and the rows of I_n are linearly independent and it is thus of full rank n.

(ii) If D is a diagonal matrix then rk(D) = #(non-zero diagonal elements in D). This follows by an argument similar to that in (i).

(iii) rk(X) = 0 if and only if X = 0. If the rank is zero then there are no linearly independent columns and so X = 0. If X = 0 then any column x_i of X is 0 and so we have $ax_i = 0$ for any a (including at least one $a \neq 0$) so X has no linearly independent columns and thus rk(X) = 0.

(iv) $rk(\lambda X) = rk(X)$ if $\lambda \neq 0$, [obviously].

2.4 Rank Factorization

2.4.1 Matrices of Rank 1

Suppose X is m×n and rk(X) = 1 then let the columns of X be $x_1,x_2,\ldots, x_n$ and suppose (with no loss in generality) that $x_1 \neq 0$. Since X has rank 1 every column x_j ($2 \leq j \leq n$) of X must be linearly dependent on x_1. So for each j, $2 \leq j \leq n$ we have $a_1x_1 + a_jx_j = 0$, or $x_j = -a_1/a_j$, noting that $a_j \neq 0$ because if $a_j = 0$ we have $a_1x_1 = 0$ which implies $a_1 = 0$ because we know $x_1 \neq 0$ but we cannot have both a_1 and a_j zero. Thus each column of X is a multiple of the first and we can write $X = (a_1x_1, a_2x_1, \ldots, a_nx_1) = x_1a'$ which is of the form xy′ where x is an m-vector and y an n-vector. Conversely if X = xy′ then $X = (y_1x, y_2x, \ldots, y_nx)$ and so all columns are linearly dependent upon on m×1 vector and so rk(X) = 1.

Thus if a matrix is of rank 1 then it can be written as xy′ for some vectors x and y (and conversely).

2.4.2. m×n Matrices of Rank r

The above is a special case of the result that any m×n matrix of rank r can be written as UV′ where U is m×r and V is n×r and each has rank r. Since X is of rank r it has r linearly independent columns, say $u_1, u_2, \ldots, u_r$ and each column x_i of X is a linear combination of these, so $x_i = \sum_{j=1}^{r} v_{ij}u_j$ for some constants v_{ij}. Letting $U = (u_1, u_2, \ldots, u_r)$ and V = (v_{ij}) and the result follows. In passing, note that if $V = (v_1, v_2, .., v_r)$ we have $x = \sum_{j=1}^{r} u_j v'_j$, a sum of r m×n matrices each of rank 1.

2.5 Rank Inequalities

2.5.1 Sum and difference

rk(X+Y) ≤ rk(X) + rk(Y) because if r = rk(X) and s = rk(Y) then let $x_1, x_2, \ldots, x_r$ be r linearly independent columns of X and $y_1, y_2, \ldots, x_s$ be s linearly independent columns of Y. Then since every column of X can be expressed as a linear combination of the x_i and likewise every column of Y in terms of the y_j every column of X+Y can be expressed in terms of a linear combination of the r+s vectors $x_1, x_2, \ldots, x_r, y_1, y_2, \ldots, x_s$. So rk(X+Y) ≤ rk(X) + rk(Y).

rk(X–Y) ≥ |rk(X) – rk(Y)| follows from above by replacing X by X–Y and noting rk(X–Y) = rk(Y–X).

Note also that rk(X–Y) = rk(X+(–Y)) ≤ rk(X) + rk(–Y) = rk(X) + rk(Y), i.e. **rk(X–Y) ≤ rk(X) + rk(Y)**

2.5.2 Products

rk(XY) ≤ min(rk(X), rk(Y)) because if $y_1, y_2, \ldots, y_r$ are a set of linearly independent columns of Y (where rk(XY) = r and presuming that these are the first r columns of Y, without loosing generality) then any column y_j of Y can be expressed as a linear combination of these r columns. If the columns of Z=XY are $z_1, z_2, \ldots$ and noting $z_j = Xy_j$ then any column z_j of Z can be expressed as a linear combination of $z_1, z_2, \ldots, z_r$ and so rk(Z) ≤ r = rk(Y).

Similarly rk(X) ≥ rk(Z) and we have rk(XY) ≤ min(rk(X), rk(Y)).

2.5.2.1 Product with orthogonal matrix

If C is an orthogonal matrix then rk(AC) = rk(A) because rk(A) = rk(ACC′) ≤ rk(AC) ≤ rk(A)

2.5.3 Sub-matrices

If A_{ij} is a sub-matrix of A then rk(A_{ij}) ≤ rk(A) because if we express A_{ij} = EAF (see §1.8.1) then rk(A_{ij}) = rk(EAF) ≤ rk(EA) ≤ rk(A).

2.6 Exercises 2

1. Let $X_1 = \begin{pmatrix} 1.3 & 9.1 \\ 1.2 & 8.4 \end{pmatrix}$, $X_2 = \begin{pmatrix} 1.2 & 9.1 \\ 1.3 & 8.4 \end{pmatrix}$, $X_3 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 9 \end{pmatrix}$,
   $X_4 = \begin{pmatrix} 1 & 2 \\ 3 & 9 \\ 2 & 1 \end{pmatrix}$, $X_5 = \begin{pmatrix} 1 & 2 & 9 \\ 2 & 1 & 3 \\ 9 & 3 & 0 \end{pmatrix}$ and $X_6 = \begin{pmatrix} 6 & 2 & 8 \\ 5 & 1 & 6 \\ 1 & 7 & 8 \end{pmatrix}$.
   a) What is the rank of each of X_1, …, X_6?
   b) Find constants a_1, a_2 & a_3 such that $a_1C_{31}+a_2C_{32}+ a_3C_{33}= 0$ where C_{3j} (j = 1, 2, 3) are the three columns of X_3.
   c) Find constants a_1, a_2 & a_3 such that $a_1R_{41}+a_2R_{42}+ a_3R_{43}= 0$ where R_{4i} (i = 1, 2, 3) are the three rows of X_4.
2. Let $X_7 = \begin{pmatrix} 4 & 5 & 6 \\ 8 & 10 & 12 \\ 12 & 15 & 18 \end{pmatrix}$ and $X_8 = \begin{pmatrix} 4 & 12 & 8 \\ 6 & 18 & 12 \\ 5 & 15 & 10 \end{pmatrix}$.
   a) Show that X_7 and X_8 are both of rank 1
   b) Find vectors a and b such that $X_7 = ab'$
   c) Find vectors u and v such that $X_8 = uv'$
3. Let $X_9 = X_3X_4$ and $X_{10} = X_4X_3$.
   a) Evaluate X_9 and X_{10} in **R.**
   b) What is the rank of X_9?
   c) What is the rank of X_{10}?

3 Determinants

3.1 Introduction and Definitions

With every n×n matrix A = (a_{ij}) there is a value |A| (or det(A)), called the ***determinant*** of A, calculated from the elements of a_{ij} of A.

3.1.1 General Examples

(i) 2×2 matrices: $\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$

(ii) 3×3 matrices:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = \begin{array}{c} a_{11}a_{22}a_{33} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} + a_{12}a_{23}a_{31} \\ +a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} \end{array}$$

$$= a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

In (ii) above the 3×3 determinant has been 'expanded' along the first row with each element in the row multiplying the determinant of the 2×2 sub-matrix obtained by deleting the row and column containing that element. Note further that the signs alternate in the expansion. In fact, the 3×3 determinant could have been 'expanded' along the first column:

$$a_{11}a_{22}a_{33} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31}$$

$$= a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{21}\begin{vmatrix} a_{12} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} + a_{31}\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix}$$

Again each element of the column multiplies the determinant of the 2×2 sub-matrix obtained by deleting the row and column containing that element, with alternating signs.

In fact, the 3×3 determinant could be 'expanded' using any row or column in the same way with signs alternating starting with a + or a – according as the row or column number is odd or even, respectively.

For example, expanding along the second row gives

$|A| = -a_{21}|A_{(21)}| + a_{22}|A_{(22)}| - a_{23}|A_{(23)}|$ where $A_{(ij)}$ is the matrix obtained by deleting row i and column j.

The quantity $c_{ij} = (-1)^{i+j}|A_{(ij)}|$ is termed the ***cofactor* of a_{ij}.**

The ***cofactor matrix of A*** is the matrix $C = (c_{ij})$. C', the transpose of C is the ***adjoint*** of A and is denoted by $A^{\#}$.

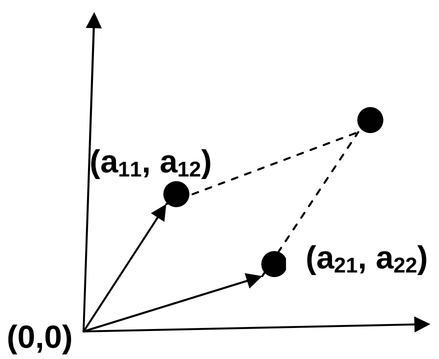
(iii) A 4×4 matrix can be evaluated by expanding it using any row or column with each element multiplying a 3×3 matrix.

(iv) In general, if A is an n×n matrix then the ***determinant*** of A is

$|A| = \sum_{k=1}^{n} a_{ik}c_{ik} = \sum_{k=1}^{n} a_{kj}c_{kj}$ for any choice of i or j.

3.1.2 Notes

(i) If (a_{11}, a_{21}) and (a_{12}, a_{22}) are the coordinates of points in a plane then $|A|$ is the volume of the parallelogram formed by the vectors $(a_{11}, a_{21})'$ & $(a_{12}, a_{22})'$. Similarly if A is a 3×3 matrix then |A| is the volume of the parallelepiped formed by the three columns of A.



(ii) More generally for an n×n matrix, |A| represents the volume of the parallelotope formed by the columns of A. In this sense |A| reflects the ***size of the matrix* A.**

(iii)★★ Note that each term in the expansion of the determinant is a product of n elements, no two of which are in the same row or column. The sign of the term depends on whether the sequence of columns [when expanding by rows] is an even or odd permutation of the integers 1, 2, …, n.

3.1.3 Numerical Examples

(i) If A= I_n then |A| = 1, expand successively by any row or column.

(ii) Let $X = \begin{pmatrix} 4 & 6 \\ 6 & 9 \end{pmatrix}$, then |X| = 4×9 – 6×6 = 0.

(iii) Let $X = \begin{pmatrix} 3 & -1 \\ 2 & 2 \end{pmatrix}$, |X| = 3×2 – (– 1)×2 = 6 + 2 = 8.

(iv) Let $X = \begin{pmatrix} 1 & 5 & 6 \\ 2 & 6 & 8 \\ 7 & 1 & 8 \end{pmatrix}$,

then |X| = 1×(6×8 – 8×1) – 2×(5×8 – 6×1) + 7×(5×8 –6×6)

= 40 – 68 + 28 = 0

(v) $X = \begin{pmatrix} 1 & -2 & 2 \\ 2 & 0 & 1 \\ 1 & 1 & -2 \end{pmatrix}$, then (expanding by the middle row)

|X| = –2×(–2×(–2) – 2) + 0 – 1×(1×1 – (–2)×1) = – 4 – 3 = –7

3.1.4 Implementation in R

The determinant of a square matrix in **R** is provided by the function `det(.)`.

3.1.4.1 Examples

```
> options(digits=3)
> det(matrix(c(1,0,0,1),2,2,byrow=T))
[1] 1
> det(matrix(c(1,0,0,0,1,0,0,0,1),3,3,byrow=T)
+ )
[1] 1
> det(matrix(c(4,6,6,9),2,2,byrow=T))
[1] -2e-15
> det(matrix(c(3,-1,2,2),2,2,byrow=T))
[1] 8
> det(matrix(c(1,5,6,2,6,8,7,1,8),3,3,byrow=T))
[1] -4.64e-14
> det(matrix(c(1,-2,2,2,0,1,1,1,-2),3,3,byrow=T))
[1] -7
```

(Note that these are the matrices considered in §3.1.3. above)

3.2 Properties of Determinants

3.2.1 Elementary Row & Column Operations on Determinants

(i) If we multiply a single row(or column) of A by a scalar λ then the determinant is multiplied by λ (follows from definition).

(ii) If we interchange two rows (columns) of a matrix A then the determinant changes sign but not absolute value (proof not given, see examples below).

(iii) If we add a scalar multiple of one row (column) to another (column) the determinant does not change (proof not given, see examples below). This is useful in evaluation of matrices.

3.2.2 Other Properties of Determinants

(i) $|A'| = |A|$ (this follows from the fact that a determinant can be expanded either by rows or by columns).

(ii) If A is $n\times n$ and λ is a scalar then $|\lambda A| = \lambda^n|A|$ (this follows directly from the definition).

(iii) If a complete row [or column] of A consists of zeroes (i.e. if $a_{ij} = 0$ for all j [or for all i] then $|A| = 0$ (consider expanding the determinant by that row [column]).

(iv) If A has two identical rows [columns] then $|A| = 0$ (replace one of the rows [columns] by the difference between the two identical rows [columns].

(v) If A is $n\times n$ and $rk(A) < n$ then $|A| = 0$ (if A is not of full rank then there is a linear combination of rows [columns] that is zero, so replace any row [column] by this linear combination. The converse is also true, i.e. if $rk(A) = n$ then $|A| \neq 0$ (see next chapter).

(vi) If $D = \text{diag}(d_1, d_2, \ldots, d_n)$, i.e. diagonal matrix with elements $d_1, d_2, \ldots, d_n$ down the diagonal then $|D| = d_1 d_2 \ldots d_n$ (expand $|D|$ successively by leftmost columns).

(vii) If T is a triangular matrix with elements $t_1, t_2, \ldots, t_n$ down the diagonal (i.e. if T is upper [lower] triangular then all elements below [above] the diagonal are zero) then $|T| = t_1 t_2 \ldots t_n$ (expand $|T|$ successively by leftmost [rightmost] columns)

(viii) $|AB| = |A||B|$ for $n \times n$ matrices A and B (proof not given, see examples below).

3.2.3 Illustrations of Properties of Determinants

(i) If $X = \begin{pmatrix} 3 & -1 \\ 2 & 2 \end{pmatrix}$ then $X' = \begin{pmatrix} 3 & 2 \\ -1 & 2 \end{pmatrix}$

so $|X'| = 3\times 2 - 2\times(-1) = 6 + 2 = 8 = |X|$

(ii) If $X = \begin{pmatrix} 3 & -1 \\ 2 & 2 \end{pmatrix}$, then

$\begin{vmatrix} 3\times 3 & -1 \\ 3\times 2 & 2 \end{vmatrix} = 3\times 3\times 2 - 3\times 2\times(-1) = 18 + 6 = 24 = 3\times 8 = 3|X|$

(iii) If $X = \begin{pmatrix} 3 & -1 \\ 2 & 2 \end{pmatrix}$, then $\begin{vmatrix} -1 & 3 \\ 2 & 2 \end{vmatrix} = -1\times 2 - 3\times 2 = -8 = -|X|$

(iv) If $X = \begin{pmatrix} 1 & -2 & 2 \\ 2 & 0 & 1 \\ 1 & 1 & -2 \end{pmatrix}$ and $Y = \begin{pmatrix} 2 & 0 & 1 \\ 1 & -2 & 2 \\ 1 & 1 & -2 \end{pmatrix}$ then (expanding by the top row)

$|Y| = 2\times(-2\times(-2) - 2) - 0 + 1\times(1\times 1 - (-2)\times 1) = 4 + 3 = 7 = -|X|$

(v) If $X = \begin{pmatrix} 1 & -2 & 2 \\ 2 & 0 & 1 \\ 1 & 1 & -2 \end{pmatrix}$ and $Y = \begin{pmatrix} 1-2\times 2 & -2 & 2 \\ 2-2\times 1 & 0 & 1 \\ 1-2\times(-2) & 1 & -2 \end{pmatrix}$ (subtracting twice the third column from the first column) then

$Y = \begin{pmatrix} -3 & -2 & 2 \\ 0 & 0 & 1 \\ 5 & 1 & -2 \end{pmatrix}$ so $|Y| = -\begin{vmatrix} -3 & -2 \\ 5 & 1 \end{vmatrix} = -7 = |X|$

(vi) If $X=\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ and $Y=\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$ then

$$XY=\begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}, \text{ so}$$

$$\begin{aligned}|XY| &= a_{11}a_{21}b_{11}b_{12}+a_{11}a_{22}b_{11}b_{22}+a_{12}a_{21}b_{21}b_{12}+a_{12}a_{22}b_{21}b_{22} \\ &\quad -a_{11}a_{21}b_{11}b_{12}-a_{21}a_{12}b_{11}b_{22}-a_{11}a_{22}b_{12}b_{21}-a_{12}a_{22}b_{21}b_{22} \\ &= a_{11}a_{22}b_{11}b_{22}-a_{11}a_{22}b_{12}b_{21}-a_{21}a_{12}b_{11}b_{22}+a_{12}a_{21}b_{21}b_{12} \\ &= (a_{11}a_{22}-a_{12}a_{21})(b_{11}b_{22}-b_{12}b_{21}) = |X||Y|\end{aligned}$$

(vii) If $X=\begin{pmatrix} 1 & \rho & \rho & \cdots & \rho \\ \rho & 1 & \rho & \cdots & \rho \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \rho & & & \ddots & \rho \\ \rho & \rho & \cdots & \rho & 1 \end{pmatrix}$ then

$$|X|=\begin{vmatrix} 1 & \rho & \rho & \cdots & \rho \\ \rho & 1 & \rho & \cdots & \rho \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \rho & & & \ddots & \rho \\ \rho & \rho & \cdots & \rho & 1 \end{vmatrix}=\begin{vmatrix} 1 & \rho & \rho & \cdots & \rho \\ \rho-1 & 1-\rho & 0 & \cdots & 0 \\ \vdots & \vdots & 1-\rho & \cdots & \vdots \\ \rho-1 & 0 & 0 & \ddots & 0 \\ \rho-1 & 0 & \cdots & 0 & 1-\rho \end{vmatrix}$$

(subtracting the first row from each of the subsequent rows)

$$=\begin{vmatrix} 1+(n-1)\rho & \rho & \rho & \cdots & \rho \\ 0 & 1-\rho & 0 & \cdots & 0 \\ \vdots & \vdots & 1-\rho & \cdots & \vdots \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1-\rho \end{vmatrix}=[1+(n-1)\rho](1-\rho)^{(n-1)}$$

(replacing the first column by the sum of all the columns and then noting the matrix is upper triangular so the determinant is the product of the diagonal elements)

3.3 Orthogonal Matrices

A is orthogonal if $AA' = A'A = I_n$, since $|A| = |A'|$ and $|I_n| = 1$ we have that if A is orthogonal then $|A| = \pm 1$. If $|A| = +1$ then A is termed a ***rotation matrix*** and if $|A| = -1$ then A is a ***reflection matrix***.

3.3.1 Examples

The matrices $A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $A_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ and $A_3 = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$ are all orthogonal and $B_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$, $B_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ and $B_3 = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ are all reflection matrices.

3.4⁺ Determinants of Partitioned Matrices

3.4.1 Some basic results

Consider the partitioned matrix $\begin{pmatrix} A & B \\ C & D \end{pmatrix}$ where the dimensions of the sub-matrices match suitably. Consider first some special cases where some of A, …, D are either 0 or identity matrices.

(i) If A and B are square matrices then [clearly] $\begin{vmatrix} A & 0 \\ 0 & D \end{vmatrix} = |A||D|$

(ii) $\begin{vmatrix} 0 & I_m \\ I_n & 0 \end{vmatrix} = (-1)^{mn}$ (obtained by column interchanges to convert this matrix to the identity matrix, each interchange changes the sign of the determinant).

(iii) If B and C are both square matrices then $\begin{vmatrix} 0 & B \\ C & 0 \end{vmatrix} = (-1)^{mn}|B||C|$,

noting $\begin{pmatrix} 0 & B \\ C & 0 \end{pmatrix} = \begin{pmatrix} B & 0 \\ 0 & C \end{pmatrix}\begin{pmatrix} 0 & I_m \\ I_n & 0 \end{pmatrix}$. It can be shewn that if B and C are not square then the matrix must be singular and so has zero determinant.

(iv) $\begin{vmatrix} I_m & B \\ 0 & I_n \end{vmatrix} = 1$ (since the matrix is [upper] triangular).

(v) Noting (if C = 0 and B ≠ 0) $\begin{pmatrix} I_m & 0 \\ 0 & D \end{pmatrix}\begin{pmatrix} I_m & B \\ 0 & I_n \end{pmatrix}\begin{pmatrix} A & 0 \\ 0 & I_n \end{pmatrix} = \begin{pmatrix} A & B \\ 0 & D \end{pmatrix}$

(and similar if B = 0 and C ≠ 0) gives $\begin{vmatrix} A & B \\ 0 & D \end{vmatrix} = \begin{vmatrix} A & 0 \\ C & D \end{vmatrix} = |A||D|$.

(vi) $\begin{vmatrix} 0 & B \\ C & D \end{vmatrix} = (-1)^m |D||BD^{-1}C|$ (where B is m×n)

because $\begin{pmatrix} I_m & -BD^{-1} \\ 0 & I_n \end{pmatrix}\begin{pmatrix} 0 & B \\ C & D \end{pmatrix} = \begin{pmatrix} -BD^{-1}C & 0 \\ C & D \end{pmatrix}$

and $|-BD^{-1}C| = (-1)^m|BD^{-1}C|$

(vii) (Proof that |AB| = |A||B|).

Let $P = \begin{pmatrix} I_n & A \\ 0 & I_n \end{pmatrix}, Q = \begin{pmatrix} A & 0 \\ -I_n & B \end{pmatrix}, R = \begin{pmatrix} 0 & AB \\ -I_n & B \end{pmatrix}$

then PQ = R, |P| = 1 and |Q| = |A||B|. Now premultiplying Q by P only multiplies the last n rows of Q by A and ads them to the first n rows, i.e. is essentially a combination of elementary row and column operations and so this leave the determinant unchanged, i.e. |PQ| = |Q| = |A||B|.

Further $|R| = (-1)^n|B||ABB^{-1}(-I_n)| = (-1)^{2n}|AB| = |AB|$.

Thus |AB| = |R| = |PQ| = |Q| = |A||B|.

3.5 A key property of determinants

The results in this section are of key importance in simplifying evaluation of the determinant of a sum of a single matrix with a product of two matrices. The proofs rely on non-obvious manipulation of partitioned matrices.

3.5.1 General result

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = |A||D - CA^{-1}B| = |D||A - BD^{-1}C|$$

(provided A [D] is non-singular) because

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = \left|\begin{pmatrix} I_m & 0 \\ -CA^{-1} & I_n \end{pmatrix}\begin{pmatrix} A & B \\ C & D \end{pmatrix}\right| = \begin{vmatrix} A & B \\ 0 & D - CA^{-1}B \end{vmatrix} \text{ and}$$

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = \left|\begin{pmatrix} I_m & -BD^{-1} \\ 0 & I_n \end{pmatrix}\begin{pmatrix} A & B \\ C & D \end{pmatrix}\right| = \begin{vmatrix} A - BD^{-1}C & 0 \\ C & D \end{vmatrix} \quad \text{(see §3.4.1(iv)}$$

3.5.1.1 Note

Replacing C by –C gives $|A||D + CA^{-1}B| = |D||A + BD^{-1}C|$

3.5.2 Important special cases

(i) Putting $A = I_m$ and $D = I_n$ in §3.5.1 above gives $|I_m - BC| = |I_n - CB|$ where B is $m \times n$ and C is $n \times m$

(ii) Similarly we have $|I_m + BC| = |I_n + CB|$

(iii) Putting C = x and B = y′ gives $|I_n + xy'| = |1 + y'x| = (1 + \Sigma x_i y_i)$

(iv) Putting B = x′ and C = –x where $x = (x_1, x_2, \ldots, x_n)'$ is a vector of length n gives $|I_n + xx'| = |I_1 + x'x| = (1 + \Sigma x_i^2)$

(v) Putting $x = 1_n$ gives $|I_n + 1_n 1_n'| = (n + 1)$

3.6 Exercises 3

1) Find the determinants of $A=\begin{pmatrix}2 & 3\\ 3 & 4\end{pmatrix}$ and $B=\begin{pmatrix}-4 & 3\\ 3 & -2\end{pmatrix}$

2) Find the determinant of $X=\begin{pmatrix}4 & 5 & 6\\ 8 & 10 & 12\\ 12 & 15 & 18\end{pmatrix}$

3) Find the determinant of $X=\begin{pmatrix}1 & 2 & 9\\ 2 & 1 & 3\\ 9 & 3 & 0\end{pmatrix}$

4) Find the determinants of $X=\begin{pmatrix}2 & 3 & 3\\ 3 & 2 & 3\\ 3 & 3 & 2\end{pmatrix}$ and $Y=\begin{pmatrix}3 & 2 & 3\\ 2 & 6 & 6\\ 3 & 6 & 11\end{pmatrix}$

5) Find the determinant of $S=\begin{pmatrix}1+\alpha & 1 & \beta\\ 1 & 1+\alpha & \beta\\ \beta & \beta & \alpha+\beta^2\end{pmatrix}$

6) Find the determinant of $S=\begin{pmatrix}2 & 1 & 3\\ 1 & 2 & 3\\ 3 & 3 & 10\end{pmatrix}$

3.5.1+ Further Exercises 3

1) [c.f. §3.2.3(vii)] If $X = \sigma^2 \begin{pmatrix} 1 & \rho & \rho & \cdots & \rho \\ \rho & 1 & \rho & \cdots & \rho \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \rho & & & \ddots & \rho \\ \rho & \rho & \cdots & \rho & 1 \end{pmatrix}$ shew that

$X = \sigma^2[(1-\rho)I_n+\rho 1_n 1_n']$ and hence

that $|X| = \sigma^{2n}(1-\rho)^{(n-1)}[1+(n-1)\rho]$

2) If $X = \begin{pmatrix} 1+\alpha & 1 & \beta \\ 1 & 1+\alpha & \beta \\ \beta & \beta & \alpha+\beta^2 \end{pmatrix}$ shew that $|X| = \alpha^2(2+\alpha+\beta^2)$ by

shewing that $X = \alpha I_3 + xx'$ for a suitable choice of x.

3) To shew $\begin{vmatrix} A & B \\ B & A \end{vmatrix} = |A+B||A-B|$:

a) Shew $\begin{pmatrix} I_n & I_n \\ 0 & I_n \end{pmatrix}\begin{pmatrix} A & B \\ A & B \end{pmatrix} = \begin{pmatrix} A+B & B+A \\ B & A \end{pmatrix}$

b) Shew $\begin{pmatrix} A+B & 0 \\ 0 & I_n \end{pmatrix}\begin{pmatrix} I_n & I_n \\ B & A \end{pmatrix} = \begin{pmatrix} A+B & B+A \\ B & A \end{pmatrix}$

c) Shew $\begin{pmatrix} I_n & I_n \\ B & A \end{pmatrix} = \begin{pmatrix} I_n & 0 \\ B & I_n \end{pmatrix}\begin{pmatrix} I_n & I_n \\ 0 & A-B \end{pmatrix}$

d) Shew $\begin{vmatrix} A & B \\ B & A \end{vmatrix} = |A+B||A-B|$

4 Inverses

4.1 Introduction and Definitions

A square n×n matrix A is ***non-singular*** if rk(A) = n; if rk(A) < n then A is ***singular***.

If A is an n×n matrix and B is also an n×n matrix such that AB = BA = I_n , the n×n identity matrix, then B is the ***inverse*** of A and is denoted by A^{-1}.

4.2 Examples:

(i) If $A=\begin{pmatrix}2&3\\3&4\end{pmatrix}$ and $B=\begin{pmatrix}-4&3\\3&-2\end{pmatrix}$ then $AB=BA=\begin{pmatrix}1&0\\0&1\end{pmatrix}=I_2$,

so $A^{-1}=\begin{pmatrix}-4&3\\3&-2\end{pmatrix}$.

(ii) $A=\begin{pmatrix}3&4\\2&3\end{pmatrix}$ and $B=\begin{pmatrix}3&-4\\-2&3\end{pmatrix}$ then $AB=BA=\begin{pmatrix}1&0\\0&1\end{pmatrix}=I_2$,

so $A^{-1}=\begin{pmatrix}3&-4\\-2&3\end{pmatrix}$.

(iii) If $A=\begin{pmatrix}1&0&0\\0&2&3\\0&3&4\end{pmatrix}$ and $B=\begin{pmatrix}1&0&0\\0&-4&3\\0&3&-2\end{pmatrix}$ then

$AB=BA=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}=I_3$ so $A^{-1}=\begin{pmatrix}1&0&0\\0&-4&3\\0&3&-2\end{pmatrix}$

(iv) if $A=\begin{pmatrix}3&0&4\\0&1&0\\2&0&3\end{pmatrix}$ and $B=\begin{pmatrix}3&0&-4\\0&1&0\\-2&0&3\end{pmatrix}$

then $AB=BA=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}=I_3$ $A^{-1}=\begin{pmatrix}3&0&-4\\0&1&0\\-2&0&3\end{pmatrix}$

(v) if $X = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and $Y = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$ (with ad – bc ≠ 0)

then $XY = YX = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I_2$ so $X^{-1} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

4.2.1 Notes

(a) Note the similarities between (i) and (iii) an also between (ii) and (iv)

(b) Note that (1) and (ii) follow from the general formula in (v) for 2×2 matrices.

4.3 Properties

4.3.1 Singular and non-singular matrices

An n×n matrix A only possess an inverse (i.e. is ***invertible***) if rk(A) = n:– If B is the inverse of A then $I_n = AB$, so $n = rk(I_n) = rk(AB) \leq \min\{rk(A), rk(B)\} \leq rk(A) \leq n$; i.e. $n \leq rk(A) \leq n$ so rk(A) = n and A is non-singular. The converse can also be proved [using ideas of vector spaces], i.e. if A is non-singular then A must possess an inverse.

4.3.2 Uniqueness

Suppose A has an inverse B and also an inverse C, then $AB = I_n$ and also $CA = I_n$, so $C = C(I_n) = C(AB) = (CA)B = (I_n)B = B$ so the inverse of A, A^{-1}, is unique.

4.3.3 Inverse of inverse

$(A^{-1})^{-1} = A$ because $(A)A^{-1} = A^{-1}(A) = I_n$ (because A^{-1} is the inverse of A, but this also means that $A(A^{-1}) = (A^{-1})A = I_n$, shewing A is the inverse of A^{-1}.

4.3.4 Determinant of Inverse

$1 = |I_n| = |AA^{-1}| = |A||A^{-1}|$, so $|A^{-1}| = |A|^{-1}$

So, if A is non-singular (i.e. rk(A) = n if A is n×n) then $|A| \neq 0$ (since A must posses an inverse if it is non-singular, §4.3.1).

4.3.5 Inverse of transpose

$(A')^{-1} = (A^{-1})'$ because $A'(A^{-1})' = (A^{-1}A)' = I_n' = I_n$ [noting that the product of transposes is the transpose of the reverse product, i.e. $X'Y' = (YX)'$].

4.3.6 Inverse of product

If A and B are both n×n non-singular matrices then $(AB)^{-1} = B^{-1}A^{-1}$ because $(AB)B^{-1}A^{-1} = AI_nA^{-1} = AA^{-1} = I_n$ and similarly $B^{-1}A^{-1}(AB) = I_n$.

4.3.6.1 Rank of product with non-singular matrix

(Generalisation of §2.5.2.1). If C is a non-singular matrix then rk(A) = rk(A) since $rk(A) = rk(ACC^{-1}) \leq rk(AC) \leq rk(A)$.

4.3.7 Orthogonal Matrices

(See also §1.7.2 and §3.3).

An n×n matrix A is orthogonal if $A^{-1} = A'$, i.e. if $AA' = A'A = I_n$. Clearly, if A is orthogonal then so is A′.

4.3.8 Scalar Products of Matrices

If A is an n×n matrix and λ is any scalar (i.e. real number or constant) then $(\lambda A)^{-1} = (1/\lambda)A^{-1}$, because $\lambda A(1/\lambda)A^{-1} = \lambda(1/\lambda)AA^{-1} = 1I_n = I_n$.

4.4 Implementation in R

The inverse of a non-singular matrix is provided in **R** by the function `solve(.)`. Two other functions in libraries will also produce inverses of non-singular matrices but they are designed to produce *generalized inverses* (see **§???** below) of non-singular matrices. These functions are `ginv(A)`(in the `MASS` library) and `MPinv(A)`(in the `gnm` library). The function `solve(.)` will generate a warning message if the matrix is singular.

4.4.1 Examples

```
> A<-matrix(c(2,3,3,4),2,2,byrow=T)
> A; solve(A)
     [,1] [,2]
[1,]    2    3
[2,]    3    4
     [,1] [,2]
[1,]   -4    3
[2,]    3   -2
> A%*%solve(A)
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

```
> A<-matrix(c(3,4,2,3),2,2,byrow=T)
> A; solve(A)
     [,1] [,2]
[1,]    3    4
[2,]    2    3
     [,1] [,2]
[1,]    3   -4
[2,]   -2    3
> A%*%solve(A)
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

```
> A<-matrix(c(1,0,0,0,2,3,0,3,4),
3,3,byrow=T)
> A; solve(A)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    2    3
[3,]    0    3    4
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0   -4    3
[3,]    0    3   -2
> A%*%solve(A)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

```
> A<-matrix(c(3,0,4,0,1,0,2,0,3),
3,3,byrow=T)
> A; solve(A)
     [,1] [,2] [,3]
[1,]    3    0    4
[2,]    0    1    0
[3,]    2    0    3
     [,1] [,2] [,3]
[1,]    3    0   -4
[2,]    0    1    0
[3,]   -2    0    3
> A%*%solve(A)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

```
> A<-matrix(c(1.3,9.1,1.2,8.4),
2,2,byrow=T)
> A; solve(A)
     [,1] [,2]
[1,]  1.3  9.1
[2,]  1.2  8.4
Error in solve.default(A) :
  system is computationally
singular: reciprocal condition
number = 1.59842e-17
```

```
> A<-matrix(c(1.2,9.1,1.3,8.4),
2,2,byrow=T)
> A; solve(A)
     [,1] [,2]
[1,]  1.2  9.1
[2,]  1.3  8.4
           [,1]       [,2]
[1,] -4.8000000  5.2000000
[2,]  0.7428571 -0.6857143
> A%*%solve(A)
            [,1]         [,2]
[1,] 1.00000e+00 1.009609e-15
[2,] 8.63025e-17 1.000000e+00
```

Note that in first example of the two immediately above the matrix is singular and the result in the second is the identity matrix I_2 to within rounding error of order 10^{-16}; see Exercises 2 in the previous chapter.

To control the number of digits printed use the function `options(.)` with `digits` specified, e.g. `options(digits=3)`.

```
> A<-matrix(c(1,2,9,2,1,3,9,3,0),
3,3,byrow=T)
> A; solve(A)
     [,1] [,2] [,3]
[1,]    1    2    9
[2,]    2    1    3
[3,]    9    3    0
       [,1]   [,2]   [,3]
[1,] -0.500  1.500 -0.167
[2,]  1.500 -4.500  0.833
[3,] -0.167  0.833 -0.167
> A%*%solve(A)
          [,1]     [,2]     [,3]
[1,]  1.00e+00 3.33e-16 8.33e-17
[2,] -5.55e-17 1.00e+00 2.78e-17
[3,]  0.00e+00 0.00e+00 1.00e+00
```

```
> A<-matrix(c(6,2,8,5,1,6,1,7,8),
3,3,byrow=T)
> A; solve(A)
     [,1] [,2] [,3]
[1,]    6    2    8
[2,]    5    1    6
[3,]    1    7    8
Error in solve.default(A) :
  system is computationally
singular: reciprocal condition
number = 7.17709e-18
```

4.5 Inverses of Patterned Matrices

If a matrix has a particular pattern then it can be the case that the inverse has a similar pattern. So, in some cases it is possible to determine the inverse by guessing the form of the inverse up to a small number of unknown constants and then determining the constants so that the product of the matrix and the inverse is the identity matrix.

For example, consider the 3×3 matrix

$X = \begin{pmatrix} 2 & 3 & 3 \\ 3 & 2 & 3 \\ 3 & 3 & 2 \end{pmatrix}$; this has identical elements down the diagonal and all off-diagonal elements are identical (but distinct from the diagonal). It is a sensible guess to look for an inverse with the same structure: $Y = \begin{pmatrix} a & b & b \\ b & a & b \\ b & b & a \end{pmatrix}$.

If XY = I_3 then we have

$$\begin{pmatrix} 2a+6b & 3a+5b & 3a+5b \\ 3a+5b & 2a+6b & 3a+5b \\ 3a+5b & 3a+5b & 2a+6b \end{pmatrix} = I_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$ so we require that

2a+6b = 1 and 3a+5b = 0, so a = –5b/3 and so b = 1/(6 – 10/3) = 3/8 and a = –5/8. Check (in R):

```
> a<- -5/8 ; b<- 3/8
> X<- matrix(c(2,3,3,3,2,3,3,3,2),3,3,byrow=T)
> Y<- matrix(c(a,b,b,b,a,b,b,b,a),3,3,byrow=T)
> X%*%Y      [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

4.5.1 Matrices of form $\alpha I_n + \beta 1_n 1_n'$

The example above is a special case of matrices of the form $\alpha I_n + \beta 1_n 1_n'$, i.e. $\begin{pmatrix} \alpha+\beta & \beta & \cdots & \beta \\ \beta & \alpha+\beta & & \vdots \\ \vdots & & \ddots & \beta \\ \beta & \cdots & \beta & \alpha+\beta \end{pmatrix}$, (in above $\alpha = -1$ and $\beta = 3$).

Numerical matrices of this form are easy to recognise. Recall that $1_n 1_n'$ is the n×n matrix with all elements equal to 1. If the inverse has a similar form $aI_n + b1_n1_n'$ then we need to find constants a and b such that $(\alpha I_n + \beta 1_n 1_n')(aI_n + b1_n1_n') = I_n$, so we need

$I_n = \alpha a I_n I_n + \alpha b I_n 1_n 1_n' + a\beta 1_n 1_n' I_n + \beta b 1_n 1_n' 1_n 1_n'$

$= \alpha a I_n + (\alpha b + a\beta + nb\beta)1_n 1_n'$ (noting $1_n'1_n = n$, see §1.9.2.1).

Thus we need $\alpha a + \alpha b + a\beta + nb\beta = 1$ and $\alpha b + a\beta + nb\beta = 0$,

so $a = \alpha^{-1}$ and $b = -\beta/\alpha(\alpha + n\beta)$.

In the numerical example above we have $a = -1$ and $b = 3/(-1+9) = 3/8$ and then the inverse is $-I_3 + 31_31_3'/8$, i.e. with diagonal elements $1 - 3/8 = 5/8$ and off-diagonal elements 3/8.

4.5.2 Matrices of form A+xy′

A further generalization of above is the result that if the n×n matrix A is non-singular and x and y are n vectors such that $y'A^{-1}x \neq -1$ then we have $(A+xy')^{-1} = A^{-1} - \frac{1}{1+y'A^{-1}x}A^{-1}xy'A^{-1}$: because

$$(A+xy')(A^{-1} - \tfrac{1}{1+y'A^{-1}x}A^{-1}xy'A^{-1}) = AA^{-1} - \tfrac{AA^{-1}xy'A^{-1}}{1+y'A^{-1}x} + xy'A^{-1} - \tfrac{xy'A^{-1}xy'A^{-1}}{1+y'A^{-1}x}$$

$$= I_n - \tfrac{I_n xy'A^{-1}}{1+y'A^{-1}x} + \tfrac{xy'A^{-1}+xy'A^{-1}y'A^{-1}x-xy'A^{-1}xy'A^{-1}}{1+y'A^{-1}x} = I_n\,.$$

Numerical matrices of this form are not easy to recognise unless the matrix A is the identity matrix or a multiple of it. It is a little easier if additionally x = y. The main use of this result is that this form arises in various theoretical developments of methodology.

4.5.2.1 Example

If $A = aI_n$ and $x = y = (x_1, x_2, \ldots, x_n)'$

then $A+xy' = \begin{pmatrix} a+x_1^2 & x_1x_2 & \cdots & x_1x_n \\ x_1x_2 & a+x_2^2 & & x_2x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1x_n & x_2x_n & \cdots & a+x_n^2 \end{pmatrix}$ which is symmetric.

For example, if a = 2 and x = (1, 2, 3)′ then $A+xx' = \begin{pmatrix} 3 & 2 & 3 \\ 2 & 6 & 6 \\ 3 & 6 & 11 \end{pmatrix}$.

Note that $A^{-1} = ½I_3$, $x'A^{-1}x = 7$ and so the formula gives the inverse as

$$\begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix} - \frac{1}{(1+7)\times 2\times 2}\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{pmatrix} = \begin{pmatrix} 0.469 & -0.063 & -0.094 \\ -.0063 & 0.375 & -0.188 \\ -0.094 & -0.188 & 0.219 \end{pmatrix}.$$

Check:

```
> X<-matrix(c(3,2,3,2,6,6,3,6,11),3,3,byrow=T)
> X
     [,1] [,2] [,3]
[1,]    3    2    3
[2,]    2    6    6
[3,]    3    6   11

> solve(X)
         [,1]    [,2]     [,3]
[1,]  0.46875 -0.0625 -0.09375
[2,] -0.06250  0.3750 -0.18750
[3,] -0.09375 -0.1875  0.21875
```

4.6⁺ Inverses of Partitioned Matrices

4.6.1 Some basic results

Consider the partitioned matrix $\begin{pmatrix} A & B \\ C & D \end{pmatrix}$ where the dimensions of the sub-matrices match suitably and non-singularity is assumed where necessary. Consider first some special cases where some of A, …, D are either 0 or identity matrices. Most of the results below can be demonstrated by direct multiplication.

Recall that $\begin{pmatrix} A & B \\ C & D \end{pmatrix}\begin{pmatrix} P & Q \\ R & S \end{pmatrix} = \begin{pmatrix} AP+BR & AQ+BS \\ CP+DR & CQ+DS \end{pmatrix}$

(i) If A and B are square matrices then [clearly]

$$\begin{pmatrix} A & 0 \\ 0 & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} & 0 \\ 0 & D^{-1} \end{pmatrix}$$

(ii) $$\begin{pmatrix} 0 & B \\ C & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & C^{-1} \\ B^{-1} & 0 \end{pmatrix}$$

(iii) $$\begin{pmatrix} 0 & B \\ B^{-1} & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & B \\ B^{-1} & 0 \end{pmatrix}$$

(iv) $\begin{pmatrix} I_m & B \\ 0 & I_n \end{pmatrix}^{-1} = \begin{pmatrix} I_m & -B \\ 0 & I_n \end{pmatrix}$

(v) $\begin{pmatrix} I_m & 0 \\ C & I_n \end{pmatrix}^{-1} = \begin{pmatrix} I_m & 0 \\ -C & I_n \end{pmatrix}$

(vi) $\begin{pmatrix} A & B \\ 0 & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} & -A^{-1}BD^{-1} \\ 0 & D^{-1} \end{pmatrix}$

(vii) $\begin{pmatrix} A & 0 \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} & 0 \\ -D^{-1}CA^{-1} & D^{-1} \end{pmatrix}$

(viii) $\begin{pmatrix} A & I_m \\ I_n & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & I_m \\ I_n & -A \end{pmatrix}$

(ix) $\begin{pmatrix} 0 & I_m \\ I_n & D \end{pmatrix}^{-1} = \begin{pmatrix} -D & I_m \\ I_n & 0 \end{pmatrix}$

(x) $\begin{pmatrix} A & B \\ C & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & C^{-1} \\ B^{-1} & -B^{-1}AC^{-1} \end{pmatrix}$

(xi) $\begin{pmatrix} 0 & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} -C^{-1}DB^{-1} & C^{-1} \\ B^{-1} & 0 \end{pmatrix}$

(xii) $\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} + A^{-1}BE^{-1}CA^{-1} & -A^{-1}BE^{-1} \\ -E^{-1}CA^{-1} & E^{-1} \end{pmatrix}$ where $E = D - CA^{-1}B$

(xiii) $\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} F^{-1} & -F^{-1}BD^{-1} \\ -D^{-1}CF^{-1} & D^{-1} + D^{-1}CF^{-1}BD^{-1} \end{pmatrix}$ where $F = A - BD^{-1}C$

4.6.1.1★★ Notes

The matrices E and F in (xii) and (xiii) above are termed the ***Schlur complements*** of A and D respectively.

4.7+ General Formulae

Let A be an n×n matrix (a_{ij}) and let C = (c_{ij}) be the cofactor matrix of A, so $C' = A^{\#}$, the adjoint of A, see §3.1.1. Then, expanding the determinant by row k we have $|A| = \sum_{j=1}^{n} a_{kj}c_{kj}$ for any k and $|A| = \sum_{j=1}^{n} a_{jk}c_{jk}$ for any column k, (these result follow from the definition of the cofactors c_{ij} in §3.1.1). Let B be the matrix obtained by replacing the k^{th} row of A by a copy of row i, then |B| = 0 since it has two identical rows. Thus, expanding |B| by this row we have $\sum_{j=1}^{n} a_{kj}c_{ij} = 0$ if $i \neq k$ and similarly $|A| = \sum_{i=1}^{n} a_{ik}c_{ij}$ if $k \neq j$, i.e. $\sum_{j=1}^{n} a_{jk}c_{ji} = \delta_{ik}|A|$ where $\delta_{ik} = 1$ or 0 as j = k or $j \neq k$. Similarly $\sum_{i=1}^{n} a_{ik}c_{ij} = \delta_{kj}|A|$

Thus $AC' = C'A = |A|\,I_n$, i.e. $AA^{\#} = A^{\#}A = |A|\,I_n$, or $A^{-1} = |A|^{-1}A^{\#}$.

4.7.1 Example

If $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ then $|A| = (a_{11}a_{22} - a_{12}a_{21})$ and $C = \begin{pmatrix} a_{22} & -a_{21} \\ -a_{12} & a_{11} \end{pmatrix}$ so $A^{\#} = \begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix}$ so $A^{-1} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix}$.

4.8 Exercises 4

1. Suppose AB = BA and that A is non-singular.

   Shew that $A^{-1}B = BA^{-1}$

2. Suppose A is an n×n orthogonal matrix and B is n×n. Shew that AB is orthogonal if, and only if, B is orthogonal.

5 Eigenanalysis of real symmetric matrices

5.1 Introduction and Definitions

In most of this chapter the matrices considered are real symmetric matrices. This restriction is with application to statistical application in mind. Details of extensions to non-symmetric and to complex matrices are readily available in the references given in 0.0.

If S is a $n\times n$ matrix then the ***eigenvalues*** of S are the roots of the ***characteristic equation*** of S, $|S-\lambda I_n| = 0$. Some authors refer to eigenvalues as characteristic values or characteristic roots.

The polynomial $p_S(\lambda)=|S-\lambda I_n|$ is called the ***characteristic polynomial*** of S. This is a polynomial of degree n and so has n roots $\lambda_1, \lambda_2, \ldots, \lambda_n$ which are not necessarily distinct. Conventionally we order these so that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$.

5.1.1 Examples

(i) $S=\begin{pmatrix}1 & 4\\ 9 & 1\end{pmatrix}$ then the characteristic equation of S is $|S-\lambda I_2|$

$$=\begin{vmatrix}1-\lambda & 4\\ 9 & 1-\lambda\end{vmatrix}=(1-\lambda)(1-\lambda)-36=\lambda^2-2\lambda-35=(\lambda-7)(\lambda+5)$$

so $\lambda_1 = 7$ and $\lambda_2 = -5$

(ii) If $S=\begin{pmatrix}5 & 3\\ 3 & 2\end{pmatrix}$ then the characteristic equation of S is $|S-\lambda I_2|$

$$=\begin{vmatrix}6-\lambda & 2\\ 2 & 2-\lambda\end{vmatrix}=(6-\lambda)(2-\lambda)-4=\lambda^2-8\lambda+8=(\lambda-4)^2-8$$

so $\lambda_1 = 4+2\sqrt{2}$ and $\lambda_2 = 4-2\sqrt{2}$.

(iii) If $S = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$ then $|S - \lambda I_3| = \lambda^3 - 6\lambda^2 + 9\lambda - 4 = (\lambda-4)(\lambda-1)^2$,

so $\lambda_1 = 4$ and $\lambda_2 = \lambda_3 = 1$, i.e. the three eigenvalues are 4 (with multiplicity 1 and 1 with multiplicity 2.

5.2 Eigenvectors

If λ is an eigenvalue of S ($n \times n$ say) then $|S - \lambda I_n| = 0$ so $A = S - \lambda I_n$ is a singular matrix and so there is a linear combination of the n columns of S equal to zero (i.e. the columns must be linearly dependent), i.e. there are constants $x_1, x_2, \ldots, x_n$, not all zero, such $A(x_1, x_2, \ldots, x_n)' = 0$, i.e. such that $Ax = 0$ or $Sx = \lambda x$. The vector x is termed the ***eigenvector*** [corresponding to the eigenvalue λ]. Since there are n eigenvalues of S, $\lambda_1, \lambda_2, \ldots, \lambda_n$ (not necessarily distinct) there are n eigenvectors $x_1, x_2, \ldots, x_n$ corresponding to the n eigenvalues. Note that $x_i \neq 0$ for all i.

To find the eigenvectors of a matrix S 'by hand' the first step is to find the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ by finding the roots of the n-degree polynomial $|S - \lambda I_n|$ and then for each λ_i in turn solve the simultaneous linear equations $Sx_i = \lambda_i x_i$ for x_i.

5.2.1 General matrices

Strictly x is termed a *right eigenvector* if $Sx = \lambda x$ and a *left eigenvector* if $x'S = \lambda x'$. Note that necessarily S must be a square matrix for an eigenvector to be defined. If S is symmetric (and therefore necessarily square) then it is easily seen that left and right eigenvectors are identical with identical eigenvalues. Left and right eigenvectors of non-symmetric matrices have the same eigenvalues $|S - \lambda I_n| = |(S - \lambda I_n)'|$.

5.3 Implementation in R

The command for producing the eigenanalysis of a matrix is `eigen()`. This produces the eigenvalues and eigenvectors of a square matrix. If the matrix is not symmetric then the command produces the right eigenvectors. The eignvalues of the matrix S are stored in the vector `eigen(S)$values` and the eigenvectors in the n×n matrix `eigen(S)$vectors`.

5.3.1 Examples

```
> X<-matrix(c(1,4,9,1),2,2,byrow=T)
> X
     [,1] [,2]
[1,]    1    4
[2,]    9    1
```

```
> eigen(X)
$values
[1]  7 -5
$vectors
      [,1]   [,2]
[1,] 0.555 -0.555
[2,] 0.832  0.832
```

To verify the eigenequation:–

```
>  eigen(X)$values[1]* eigen(X)$vectors[,1]
[1] 3.88 5.82
>  X%*%eigen(X)$vectors[,1]
      [,1]
[1,] 3.88
[2,] 5.82
```

```
> X<-matrix(c(2,1,1,1,2,1,1,1,2),
            3,3,byrow=T)
> X
     [,1] [,2] [,3]
[1,]    2    1    1
[2,]    1    2    1
[3,]    1    1    2
```

```
> eigen(X)
$values
[1] 4 1 1
$vectors
       [,1]   [,2]   [,3]
[1,] -0.577  0.816  0.000
[2,] -0.577 -0.408 -0.707
[3,] -0.577 -0.408  0.707
```

To verify the eigenequation:–

```
>  eigen(X)$values[1]* eigen(X)$vectors[,1]
[1] -2.31 -2.31 -2.31
>  X%*%eigen(X)$vectors[,1]
      [,1]
[1,] -2.31
[2,] -2.31
[3,] -2.31
```

5.3.2 Notes

In statistical applications the eigenanalysis of the variance matrix is the basis of many statistical analyses. This matrix is necessarily symmetric and also it is positive semi-definite (see §1.10). Recalling §3.1.2(ii) that the 'size' of a matrix is reflected by its determinant, combined with the properties that the sum and product of the eigenvalues are equal to the trace and determinant of the matrix (see §5.4.4(iv)–(vi) below), means that the eigenvalues of a matrix are themselves key properties of a matrix. Further the eigenvectors associated with the eigenvalues, especially the dominant (i.e. largest) and the minor (smallest) values, give further information and provide interpretations of statistical interest, e.g. directions of dominant and minor variation in the case of the eigenanylsis of a variance matrix.

5.4 Properties of eigenanalyses

5.4.1 Properties related to uniqueness

(i) If x is an eigenvector of S then any scalar multiple kx is also an eigenvector because $S(kx) = \lambda kx$. Usually eigenvectors are normalized so that $x'x = 1$ which means that an eigenvector is determined up to its sign.

(ii) If λ_i and λ_j are distinct eigenvalues of S with eigenvectors x_i and x_j then x_i and x_j are distinct: suppose $x_i = x_j = x$ say, then we have $Sx = \lambda_i x$ and $Sx = \lambda_j x$ so $(\lambda_I - \lambda_j)x = 0$, so $x = 0$ because $\lambda_I \neq \lambda_j$ which contradicts x being an eigenvector.

Note that this is true whether or not S is symmetric.

(iii) If x_i and x_j are distinct eigenvectors of S with the same eigenvalue λ then any linear combination of x_i and x_j is also an eigenvector of S since $Sx_i = \lambda x_i$ and $Sx_j = \lambda x_j$ then $S(a_1x_i + a_2x_j) = \lambda(a_1x_i + a_2x_j)$. For example, the identity matrix I_n has every vector x as an eigenvector.

5.4.2 Properties related to symmetric matrices

(i) Suppose now that S is real and symmetric, i.e. $S = S'$, then the eigenvalues of S are *real*, since suppose λ_j and x_j are the eigenvalues and vectors and that $\lambda_j = \mu_j + i\nu_j$, $x_j = y_j + iz_j$ (where here $i = (-1)^{½}$), then equating real and imaginary parts of $Sx_j = \lambda_j x_j$ gives $Sy_j = \mu_j y_j - \nu_j z_j$, and $Sz_j = \nu_j y_j + \mu_j z_j$. Premultiplying the first equation by z_j' and the second by y_j' and noting $z_j'Sy_j = (z_j'Sy_j)'$ (since it's a scalar) $= y_j'Sz_j$ (since S is symmetric by presumption) and subtracting the two equations gives $\nu_j z_j'z_j + \nu_j y_j'y_j = 0$, so $\nu_j = 0$ because $z_j'z_j + y_j'y_j > 0$, i.e. λ_j is real.

(ii) If S is symmetric then eigenvectors corresponding to distinct eigenvalues are orthogonal because if $Sx_i = \lambda_i x_i$ and $Sx_j = \lambda_j x_j$ then $x_j'Sx_i = \lambda_i x_j'x_i$ and $x_i'Sx_j = \lambda_j x_i'x_j$ but $x_j'Sx_i = (x_j'Sx_i)'$ because it is a scalar (see §1.9.2.1) $(x_j'Sx_i)' = x_i'S'x_j = x_i'Sx_j$ (see §1.12.3.5 and noting S is symmetric), so $(\lambda_i - \lambda_j)x_i'x_j$ (noting $x_i'x_j = x_j'x_i$) and since $\lambda_i \neq \lambda_j$ we have $x_i'x_j = 0$ and thus x_i and x_j are orthogonal.

5.4.2.1 Left and right eigenvectors of non-symmetric matrices

(i) If x_i is a right eigenvector of the $n\times n$ matrix A with eigenvalue λ_i then $Ax_i = \lambda_i x_i$ so $x_i'A' = \lambda_i x_i'$ and so x_i' is a left eigenvector of A′. Similarly, a right eigenvector y_i of A′ is a left eigenvector of A.

(ii) If x_i and y_j' are left and right eigenvectors of A corresponding to distinct eigenvalues λ_i and λ_j then x_i and y_j are orthogonal because we have $Ax_i = \lambda_i x_i$ and $y_j'A = \lambda_i y_j'$. Pre-multiplying the first by y_j' and post-multiplying the second by x_i and subtracting gives $(\lambda_i - \lambda_j)y_j'x_i = 0$ and so y_j and x_i are orthogonal for $i \neq j$. The vectors x_i and y_i can be standardized so that $x_i'y_i = 1$ in which case the left and roiight eigenvectors are said to be ***biorthogonal.*** Note that neither x_i not the y_j are themselves orthogonal unless the matrix A is symmetric.

5.4.2.2 Illustration of biorthogonality

```
> X<-matrix(c(1,4,9,1),2,2,byrow=T)
> X
     [,1] [,2]
[1,]    1    4
[2,]    9    1
```

```
> eigen(X)
$values
[1]  7 -5
$vectors
      [,1]   [,2]
[1,] 0.555 -0.555
[2,] 0.832  0.832
```

```
> eigen(t(X))
$values
[1]  7 -5
$vectors
      [,1]   [,2]
[1,] 0.832 -0.832
[2,] 0.555  0.555
```

```
> t(eigen(X)$vectors[,1])%*%eigen(t(X))$vectors[,2]
         [,1]
[1,] -4.5e-17
```

5.4.3 Properties related to functions of matrices

(i) If x and λ are an eigenvector and an eigenvalue of S and if S is non-singular then X is an eigenvector of S^{-1} with eigenvalue λ^{-1} since if $Sx = \lambda x$ then $S^{-1}Sx = \lambda S^{-1}x$ so $S^{-1}x = \lambda^{-1}x$ shewing x is an eigenvector of S^{-1} with eigenvalue λ^{-1}.

(ii) If x and λ are an eigenvector and an eigenvalue of S then x is an eigenvector of S^k with eigenvalue λ^k since $S^kx = S^{k-1}(Sx) = S^{k-1}(\lambda x) = \ldots = \lambda^k x$.

(iii) If x and λ are an eigenvector and an eigenvalue of S then x and $(a-b)\lambda$ are an eigenvector and eigenvalue of $aI_n - bS$ since $(aI_n - bS)x = ax - b\lambda x = (a-b)\lambda x$.

(iv) If S is $n\times m$ and T is $m\times n$ where $n \geq m$ then ST and TS have the same non-zero eigenvalues. The eigenvalues of ST are the roots of $|ST - \lambda I_n| = 0$ but $|ST - \lambda I_n| = (-\lambda)^{n-m}|TS - \lambda I_m|$, see §3.5.2(i). Note that this implies that ST has at most $n-m$ non-zero eigenvalues.

(v) If x and λ are an eigenvector and an eigenvalue of ST then Tx is an eigenvector of TS corresponding to eigenvalue λ because we have $STx = \lambda x$ so $TS(Tx) = \lambda(Tx)$.

(vi) If X is an $m\times n$ matrix then XX′ and X′X have the same non-zero eigenvalues. This follows directly from (iv) above.

(vii) If x and λ are an eigenvector and an eigenvalue of S and T is a non-singular $n\times n$ matrix then λ is an eigenvalue of TST^{-1} corresponding to eigenvector Tx because if $Sx = \lambda x$ then $(TST^{-1})Tx = \lambda Tx$.

5.4.4 Properties related to determinants

(i) If S is diagonal [or triangular] then the eigenvalues are the diagonal elements since $S-\lambda I_n$ is diagonal [or triangular] and the determinant of a diagonal [or triangular] matrix is the product of its diagonal elements (see §3.2.2)

(ii) S is non-singular if and only if all of its eigenvalues are non-zero since $0 = |S-\lambda I_n| = |S|$ if $\lambda = 0$ and if $|S| = 0$ then $\lambda = 0$ satisfies $|S-\lambda I_n| = 0$ and so is an eigenvalue of S.

(iii) If μ is **not** an eigenvalue of S then $S - \mu I_n$ is non-singular since if $|S - \mu I_n| = 0$ then μ would be an eigenvalue of S.

(iv) If S has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ then $|S| = \prod_{i=1}^{n}\lambda_i$ because the λ_i are the n roots of $|S - \lambda I_n| = 0$,

so $|S - \lambda I_n| = (\lambda_1-\lambda)(\lambda_2-\lambda)\ldots\ldots(\lambda_n-\lambda)$ and putting $\lambda = 0$ gives the result.

(v) If S has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ then $\mathrm{tr}(S) = \sum_{i=1}^{n}\lambda_i$, comparing the coefficients of λ^{n-1} in

$|S - \lambda I_n| = (\lambda_1-\lambda)(\lambda_2-\lambda)\ldots\ldots(\lambda_n-\lambda)$.

(vi) If S has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ then $\mathrm{tr}(S^k) = \sum_{i=1}^{n}\lambda_i^k$ which follows from (v) and §5.2.1.3(ii)

5.4.5 Properties related to diagonalisation

(i) If X is an $n\times n$ matrix with distinct eigenvalues $\lambda_1 > \lambda_2 > \ldots > \lambda_n$ then there exists a non-singular matrix T and diagonal matrix Λ such that $T^{-1}XT = \Lambda$ and the elements of Λ are the λ_i:–

If the eigenvectors of X are x_i, $i = 1, 2, \ldots, n$ then $Xx_i = \lambda_i x_i$ $i = 1, 2, \ldots, n$ and if $T = (x_1, x_2, \ldots, x_n)$ (i.e. the matrix composed of the n eigenvectors as columns) then T is non-singular since the eigenvectors are linearly independent. Further $XT = (\lambda_1 x_1, \lambda_2 x_2, \ldots, \lambda_n x_n) = \Lambda T$ where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ so, multiplying by T^{-1} we have $T^{-1}XT = \Lambda$.

(ii) If X is an $n\times n$ matrix with distinct eigenvalues $\lambda_1 > \lambda_2 > \ldots > \lambda_n$ and Y commutes with X then $T^{-1}XT = M$ for some diagonal matrix M because if $Xx_i = \lambda_i x_i$ then $YXx_i = \lambda_i Yx_i$ so $X(Yx_i) = \lambda_i(Yx_i)$ shewing that Yx_i is another eigenvector of X corresponding to λ_I but the λ_I are distinct so Yx_i must be a scalar multiple of x_i, i.e. $Yx_i = \mu_i x_i$ for some scalar μ_i. Thus x_i is an eigenvector of Y and thus $XT = TM$ where $M = \text{diag}(\mu_1, \mu_2, \ldots, \mu_n)$.

(iii) If S is a symmetric matrix then there exists an orthogonal matrix T and a diagonal matrix Λ such that $T'ST = \Lambda$.

This result requires a substantial proof in the general case. In the case where the eigenvalues of S are distinct then it follows from (i) since by choosing the eigenvectors to be normalised to ensure $x_i'x_i = 1$ we can ensure T is orthogonal so $T^{-1} = T'$.

In the general case where there are some multiple eigenvalues (possibly some of which may be zero) we need to choose k orthogonal eigenvectors corresponding to an eigenvalue with multiplicity k. The most straightforward proof that this is possible is by induction and can be found in the references cited.

(iv) If S is a symmetric matrix then we can write S as $T\Lambda T' = S$ where Λ is the diagonal matrix of eigenvalues of S and T is the matrix of eigenvectors.

(v) If X and Y are both symmetric and if X and Y commute then result (ii) above can be generalised to there are diagonal matrices Λ and M and a matrix T such that $T'XT = \Lambda$ and $T'YT = M$. If we have $T'XT = \Lambda$ and $T'YT = M$ then $XY = T\Lambda T'TMT' = T\Lambda MT' = TM\Lambda T' = TMT'T\Lambda T' = YX$, noting diagonal matrices commute. The proof of the converse is more difficult and is not given here. In the particular case that the eigenvalues are distinct the result follows from arguments similar to that in (ii) noting that the eigenvectors in T can be chosen to be orthogonal.

5.4.6 Properties related to values of eigenvalues

(i) (Upper & lower bounds for a Rayleigh quotient $\frac{x'Sx}{x'x}$).

If S is an $n\times n$ symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and x is any vector then $\lambda_1 \geq \frac{x'Sx}{x'x} \geq \lambda_n$. This follows by noting that $x'Sx = x'T\Lambda T'x = y'\Lambda y = \Sigma_j\lambda_j y_j^2$, where $y = T'x$ so $\lambda_1\Sigma_j y_j^2 \geq x'Sx \geq \lambda_n\Sigma_j y_j^2$ and $\Sigma_j y_j^2 = y'y = x'TT'x = x'x$ since T is orthogonal.

(ii) S is positive definite (i.e. $S > 0$) if and only if all the eigenvalues of S are strictly positive: If all the λ_i are positive then in particular $\lambda_n > 0$ so $x'Sx \geq \lambda_n\Sigma_j y_j^2 > 0$ for any x and thus $S > 0$. Conversely, if $S > 0$ then $x'Sx > 0$ for any x. In particular we have $Sx_n = \lambda_n x_n$ so $x'Sx_n = \lambda_n x_n'x_n > 0$ so $\lambda_n > 0$ since $x_n'x_n > 0$.

(iii) S is positive semi-definite if and only if all the eigenvalues of S are non-negative. The proof of this is similar to that in (ii) above.

(iv) If S is positive definite then it is non-singular since its determinant (equal to the product of its eigenvalues) is strictly positive.

(v) If S is positive semi-definite and non-singular then it must be positive definite since its determinant (equal to the product of its eigenvalues) is strictly positive and so all its eigenvalues must be strictly positive.

5.4.7 Rank and non-zero eigenvalues

The rank of a **symmetric** matrix S is equal to the number of non-zero eigenvalues. We have rk(T′ST) = rk(Λ) = number of non-zero diagonal elements of Λ, noting §2.5.2.1 and §2.3(ii).

Note that this result is not true in general for non-symmetric matrices.

5.4.7.1 Example of non-symmetric matrix

Consider the matrix $A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & \vdots \\ 0 & 0 & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 1 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$ which has all eigenvalues equal to zero since it is a triangular matrix with all diagonal elements zero but it is clearly of rank $n-1$ since the first $n-1$ rows are linearly independent and the last row is entirely composed of zeroes. Matrices of this form are termed ***Jordan matrices***.

5.5 Decompositions

5.5.1 Spectral decomposition of a symmetric matrix

If S is a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ we have $S = T\Lambda T'$, with $T'T = I_n$ (§5.4.5(iv)). This is known as the ***spectral decomposition*** of S. It is often expressed in the form of a sum of rank 1 matrices: If $T = (x_1, x_2, \ldots, x_n)$ then $T\Lambda = (\lambda_1 x_1, \lambda_2 x_2, \ldots, \lambda_n x_n)$ and so $T\Lambda T' = \Sigma_j \lambda_j x_j x_j'$. Each of the matrices $x_j x_j'$ is of rank 1. If there are r non-zero eigenvalues (so $\lambda_{r+1} = \ldots\ldots = \lambda_n = 0$) then the summation is over r terms.

5.5.1.1 Square root of positive semidefinite matrices

If $S \geq 0$ then all of its eigenvalues are non-negative, i.e. $\lambda_i \geq 0$ and also we have $S = T\Lambda T'$, with $T'T = I_n$ and $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. Define $\Lambda^{½}$ by $\Lambda^{½} = \text{diag}(\lambda_1^{½}, \lambda_2^{½}, \ldots, \lambda_n^{½})$. Then let $S^{½} = T\Lambda^{½}T'$ and $(S^{½})^2 = T\Lambda^{½}T'T\Lambda^{½}T' = T\Lambda^{½}\Lambda^{½}T' = T\Lambda T' = S$ and so $S^{½}$ is a square root of S. Note that there are many other matrices Q such that $Q^2 = S$ but $S^{½}$ is the only one such that T is orthogonal. Other [positive] powers of S can be defined similarly. If S is positive definite then negative powers can also be defined in a similar way.

5.5.2 Singular value decomposition (svd) of an m×n matrix

If A is an m×n matrix with $\text{rk}(A) = r$ [$\leq \min\{m, n\}$] then there are orthogonal matrices U and V and a diagonal matrix Λ such that $A = U\Lambda^{½}V'$. The elements $\lambda^{½}$ are called the ***singular values*** of the matrix A.

5.5.2.1★★ Proof of svd

To prove this first note that AA′ and A′A are both positive semidefinite since $x'AA'x = (A'x)'(A'x) \geq 0$ (likewise A′A) and they have the same non-zero eigenvalues, see §5.4.2.(iv), $\lambda_1, \lambda_2, \ldots, \lambda_r$ say. If we define U to be the m×r matrix of eigenvectors of AA′ corresponding to the non-zero eigenvalues, so $AA'U = U\Lambda$ and the m×r matrix U_2 to be chosen so that it is orthogonal and $AA'U_2 = O$, i.e. the columns of U_2 corresponds to the zero eigenvalues of AA′, then $UU' + U_2U_2' = I_m$.

Define $V = A'U\Lambda^{-½}$. Then we have $A'AV = V\Lambda$ and $V'V = I_r$.

Since $AA'U_2 = O$ we must have $A'U_2 = O$, see 1.13.1★★(b),

so $A = I_mA = (UU' + U_2U_2')A = UU'A = UI_rU'A = U\Lambda^{½}\Lambda^{-½}U'A = U\Lambda^{½}V'$.

5.5.2.2 Note

Note that U and V are eigenvectors of AA′ and A′A, each with eigenvalues Λ.

5.5.3 Implementation of svd in R

The command for producing the singular value decomposition of a matrix is `svd()`.

5.5.3.1 Examples

```
> options(digits=3)
```

A symmetric matrix:–

```
> S<-matrix(c(2,1,1,1,2,1,1,1,3)
                      ,3,3)
> S
     [,1] [,2] [,3]
[1,]    2    1    1
[2,]    1    2    1
[3,]    1    1    3
> eigen(S)
$values
[1] 4.41 1.59 1.00

$vectors
       [,1]   [,2]   [,3]
[1,] -0.500 -0.500  0.707
[2,] -0.500 -0.500 -0.707
[3,] -0.707  0.707    0

> svd(S)
$d
[1] 4.41 1.59 1.00

$u
       [,1]   [,2]      [,3]
[1,] -0.500  0.500  0.707
[2,] -0.500  0.500 -0.707
[3,] -0.707 -0.707    0

$v
       [,1]   [,2]   [,3]
[1,] -0.500  0.500  0.707
[2,] -0.500  0.500 -0.707
[3,] -0.707 -0.707    0
```

```
> eigen(S%*%t(S))
$values
[1] 19.49  2.51  1.00

$vectors
       [,1]   [,2]   [,3]
[1,] -0.500  0.500  0.707
[2,] -0.500  0.500 -0.707
[3,] -0.707 -0.707    0

> eigen(t(S)%*%S)
$values
[1] 19.49  2.51  1.00

$vectors
       [,1]   [,2]   [,3]
[1,] -0.500  0.500  0.707
[2,] -0.500  0.500 -0.707
[3,] -0.707 -0.707    0
```

A nonsymmetric matrix:–

```
> A<-matrix(c(2,1,1,1,2,1,4,2,3),
                              3,3)
> A
     [,1] [,2] [,3]
[1,]    2    1    4
[2,]    1    2    2
[3,]    1    1    3
> eigen(A)
$values
[1] 5.45 1.00 0.55

$vectors
      [,1]   [,2]   [,3]
[1,] 0.716  0.707 -0.925
[2,] 0.494 -0.707  0.268
[3,] 0.494    0    0.268

> svd(A)
$d
[1] 6.265 1.269 0.377

$u
       [,1]   [,2]   [,3]
[1,] -0.725  0.458 -0.514
[2,] -0.444 -0.882 -0.160
[3,] -0.526  0.113  0.843

$v
       [,1]   [,2]    [,3]
[1,] -0.386  0.117 -0.9149
[2,] -0.342 -0.940  0.0245
[3,] -0.857  0.322  0.4028
```

```
> eigen(A%*%t(A))
$values
[1] 39.248  1.610  0.142

$vectors
       [,1]   [,2]   [,3]
[1,] -0.725  0.458  0.514
[2,] -0.444 -0.882  0.160
[3,] -0.526  0.113 -0.843

> eigen(t(A)%*%A)
$values
[1] 39.248  1.610  0.142

$vectors
       [,1]   [,2]    [,3]
[1,] -0.386  0.117  0.9149
[2,] -0.342 -0.940 -0.0245
[3,] -0.857  0.322 -0.4028
```

A nonssquare matrix:–

```
> B<-matrix(c(1,2,3,4,5,6),2,3)
> B
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> svd(B)
$d
[1] 9.526 0.514

$u
       [,1]   [,2]
[1,] -0.620 -0.785
[2,] -0.785  0.620

$v
       [,1]   [,2]
[1,] -0.230  0.883
[2,] -0.525  0.241
[3,] -0.820 -0.402
```

```
> eigen(B%*%t(B))
$values
[1] 90.735  0.265

$vectors
      [,1]   [,2]
[1,] 0.620 -0.785
[2,] 0.785  0.620

> eigen(t(B)%*%B)
$values
[1]  9.07e+01  2.65e-01 0

$vectors
       [,1]   [,2]   [,3]
[1,] -0.230  0.883  0.408
[2,] -0.525  0.241 -0.816
[3,] -0.820 -0.402  0.408
```

A Jordan matrix:–

```
> j<-matrix(c(0,1,0,0,0,1,0,0,0),3,3)
> j
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    1    0    0
[3,]    0    1    0
> eigen(j)
$values
[1] 0 0 0

$vectors
     [,1]  [,2] [,3]
[1,]    0     0    0
[2,]    0     0    0
[3,]    1    -1    0
```

```
> svd(j)
$d
[1] 1 1 0

$u
     [,1] [,2] [,3]
[1,]    0    0    1
[2,]    0   -1    0
[3,]   -1    0    0

$v
     [,1] [,2] [,3]
[1,]    0   -1    0
[2,]   -1    0    0
[3,]    0    0    1

>
```

5.6 Eigenanlysis of matrices with special structures

To show that a vector x and scalar λ are an eigenvector and eigenvalue of a matrix it is only necessary to demonstrate that $Ax = \lambda x$. Sometimes the matrix A has a particular structure that can be manipulated, perhaps using the 'useful tricks' indicated in §1.9.2. Sometimes this may be deceptively simple and not obvious without experience. Particular results relate to rank 1 matrices and matrices composed as the sum of a rank 1 matrix with a scalar multiple of the identity matrix.

5.6.1 The rank one matrix xx′

If x is a vector of length n then xx' is an $n\times n$ matrix of rank 1 (since $\text{rk}(xx') < \min\{\text{rk}(x), \text{rk}(x')\} = 1$). We have $xx'x = x(x'x) = (x'x)x$, noting that $(x'x)$ is a scalar and so commutes with the vector x. This is in the form $Ax = \lambda x$ with $A = xx'$ and $\lambda = (x'x)$ and so x is an eigenvector of xx' with eigenvalue $(x'x)$. Since xx' is symmetric and of rank 1 it has only one non-zero eigenvalue.

5.6.2 The rank one matrix Sxx′

If x is a vector of length n then Sxx′ is an n×n matrix of rank 1 (since rk(xx′) < min{rk(S), rk(xx′)} = 1). We have Sxx′Sx = Sx(x′Sx) = (x′Sx)Sx, noting that (x′Sx) is a scalar and so commutes with the vector Sx. This is in the form $Ax = \lambda x$ with $A = Sxx'$ and $\lambda = (x'Sx)$ and so Sx is an eigenvector of Sxx′ with eigenvalue (x′Sx). Note that Sxx′ has only one non-zero eigenvalue since it is symmetric and of rank 1.

5.6.2 The matrix aI_n + bxy′

$(aI_n + bxy')x = aI_nx + bxy'x = ax + b(y'x)x$ (noting that y′x is a scalar and so commutes with x). Thus $(aI_n + bxy')x = (a + by'x)x$ and thus x is an eigenvector of $aI_n + bxy'$ with eigenvalue $(a + by'x)$. The rank of $aI_n + bxy'$ is not in general 1 (e.g. consider $a = 1$ and $b = 0$) and so will in general have other non-zero eigenvalues and corresponding non-trivial eigenvectors.

To find the other eigenvalues consider $|aI_n + bxy' - \lambda I_n| = |(a-\lambda)I_n + bxy'|$
$= (a-\lambda)^n|I_n + bxy'/(a-\lambda)| = (a-\lambda)^n|1 + by'x/(a-\lambda)|$, see §3.5.2(iii)
$= (a-\lambda)^{n-1}(a + by'x - \lambda)$ and so the other eigenvalues are a with multiplicity $n-1$. Note that if $x = y$ and $a \neq 0$ and $a + by'x \neq 0$ then the matrix is symmetric with n non-zero eigenvalues and so is of full rank and thus non-singular.

5.7 Summary

If A is a real n×n matrix.

The eigenvalues of A are the roots of the n-degree polynomial in λ:

$$q(\lambda)= \det(A - \lambda I_n)=0 \ldots\ldots\ldots\ldots\ldots\ldots * $$

Let these be $\lambda_1,\lambda_2,\ldots\ldots\lambda_n$. Then, since the coefficient of λ^n in equation $*$ is $(-1)^n$ we have $q(\lambda) = \prod_{i=1}^{n}(\lambda_i - \lambda) \ldots\ldots\ldots\ldots\ldots **$

1. Comparing coefficients of λ^{n-1} in $*$ and $**$ gives

$$\sum_{i=1}^{n}\lambda_i = \mathrm{trace}(A) = \sum_{i=1}^{n} a_{ii}$$

2. Putting λ=0 in $*$ and $**$ gives

$$\prod_{i=1}^{n}\lambda_i = \det(A) = |A|$$

   Since the matrices $A - \lambda_i I_n$ are singular (i.e. have zero determinant) there exist vectors x_i called the ***eigenvectors*** of A such that

$$(A - \lambda_i I_n)x_i = 0, \text{ i.e. } Ax_i - \lambda_i x_i = 0.$$

   [Strictly, if A is non-symmetric, the x_i are right-eigenvectors and we can define left-eigenvectors y_i such that $y_i'A - \lambda_i y_i' = 0$]

3. Suppose C is any n×n non-singular square matrix, since $|A - \lambda I_n| = |C||A - \lambda I_n||C^{-1}| = |CAC^{-1} - \lambda I_n|$ we have:

   A and CAC^{-1} have the same eigenvalues.

4. If $Ax_i=\lambda_i x_i$ then $(CAC^{-1})(Cx_i)=\lambda_i(Cx_i)$ so the eigenvectors of CAC^{-1} are Cx_i

5. If A is n×m and B is p×n then

   $|AB - \lambda I_n| = (-\lambda)^{n-m}|BA - \lambda I_m|$ so the non-zero eigenvalues of AB and BA are identical.

6. Since, if $ABx_i = \lambda x_i$ then $(BA)(Bx_i) = \lambda(Bx_i)$, we have that the eigenvectors of BA are obtained by premultiplying those of AB by B.

7. Suppose now that A is symmetrical, i.e. A = A′, we can show that the eigenvalues of A are *real*, since suppose λ_i and x_i are the eigenvalues and vectors and that $\lambda_j = \mu_j + i\nu_j$, $x_j = y_j + iz_j$ then equating real and imaginary parts of $Ax_j = \lambda_j x_j$ gives
   $Ay_j = \mu_j y_j - \nu_j z_j$ ....∗∗∗ and $Az_j = \nu_j y_j + \mu_j z_j$ ....∗∗∗∗
   Premultiplying ∗∗∗ by z_j' and ∗∗∗∗ by y_j' and noting $z_j'Ay_j = (z_j'Ay_j)'$ (since it's a scalar) $= y_j'A'_j = y_j'Az_j$ (since A is symmetric by presumption) and subtracting the two equations gives the result.
8. Suppose again A is symmetric and that λ_j and λ_k are distinct eigenvalues with corresponding eigenvectors x_j and x_k. Then $Ax_j = \lambda_j x_j$ and $Ax_k = \lambda_k x_k$. Premultiplying these by x_k' and x_j' respectively and noting that $x_j'Ax_k = x_k'Ax_j$ since A is symmetric gives
   $(\lambda_j - \lambda_k)x_j'x_k = 0$; since $\lambda_j \neq \lambda_k$ (by presumption) gives $x_j'x_k = 0$,
   i.e. eigenvectors with distinct eigenvalues are orthogonal.

5.7.1 Summary of key results

$n\times n$ matrix A with eigenvalues $\lambda_1,\ldots,\lambda_n$ and [right] eigenvectors $x_1,\ldots,x_n$ then

1. $\sum_{i=1}^{n}\lambda_i = \text{trace}(A)$	5. AB and BA have identical non-zero eigenvalues.
2. $\prod_{i=1}^{n}\lambda_i = \det\lvert A\rvert$	6. Eigenvectors of BA = B ∗ those of AB
3. A and CAC^{-1} have identical eigenvectors for C non-singular	7. A symmetric ⇒ eigenvalues real
4. Eigenvalues of CAC^{-1} are Cx_i	8. A symmetric ⇒ eigenvectors corresponding to distinct eigenvalues are orthogonal.

5.8 Exercises 5

1.

2. Find the eigenvalues of $X = \begin{pmatrix} 1 & \rho & \rho & \cdots & \rho \\ \rho & 1 & \rho & \cdots & \rho \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \rho & & & \ddots & \rho \\ \rho & \rho & \cdots & \rho & 1 \end{pmatrix}$

3.

6 Vector and Matrix Calculus

6.1 Introduction

This section considers various simple cases of differentiation of scalar-valued, vector-valued and matrix-valued functions of scalars, vectors and matrices with respect to scalars, vectors and matrices. For example, the quadratic form $x'Ax$ is a scalar function of a vector x, xx' is a matrix function of a vector x, tr(X) and |X| are scalar functions of a matrix X, Ax is a vector function of a vector x. Not all combinations of these will be covered here.

Broadly, the procedure consists of [partially] differentiating each element of the function with respect to each element of the arguments and arranging the results in a vector or matrix as appropriate. We use 'partial differentiation' if the argument is not a scalar; this is equivalent to differentiation with respect to the individual elements. Thus the result of differentiating a scalar with respect to a vector [matrix] will consist of the vector [matrix] of partial derivatives of the scalar with respect to the elements of the vector [matrix]. Differentiating a vector-valued function of a vector argument with respect to another vector will result in a matrix where the individual elements are the partial derivatives of each element of the function with respect to each element of the vector argument. Generally, differentiating an $m\times n$ matrix with respect to a $p\times q$ matrix can be defined and will result in a matrix of dimension $mp\times nq$. We consider only the cases where not only is one of m and p equal to 1 but also one of n and q equals 1. Other cases can be handled with the use of Kronecker products. This is beyond the scope of these notes.

Most of the basic results can be obtained by expressing the functions of the vectors in terms of the individual elements and expanding, for example, inner products of vectors as sums of products of individual elements. It may help understanding to write out explicitly the cases n = 1 and n = 2. Many of the basic rules of scalar calculus of single variables (e.g. differentiation of products etc) carry through recognisably to the vector and matrix cases.

6.2 Differentiation of a scalar with respect to a vector

If x is a vector of length n and f = f(x) is a scalar function of x then $\frac{\partial f}{\partial x}$ is ***defined*** to be the vector $\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$.

6.2.1 Derivative of a′x

$$f(x) = a'x = \Sigma_j a_j x_j \text{ so } \frac{\partial f}{\partial x_i} = \frac{\partial(\Sigma_j a_j x_j)}{\partial x_i} = a_i, \text{ so } \frac{\partial f}{\partial x} = \frac{\partial(a'x)}{\partial x} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = a.$$

6.2.2 Derivative of x′x

$$f(x) = x'x = \Sigma_j x_j^2 \text{ so } \frac{\partial f}{\partial x_i} = \frac{\partial(\Sigma_j x_j^2)}{\partial x_i} = 2x_i, \text{ so } \frac{\partial f}{\partial x} = \frac{\partial(x'x)}{\partial x} = \begin{pmatrix} 2x_1 \\ 2x_2 \\ \vdots \\ 2x_n \end{pmatrix} = 2x.$$

6.2.3 Derivative of quadratic forms x′Sx

We can, without loss of generality, take S to be symmetric (see §1.10).

First consider special cases of n = 1 and n = 2:

Case n = 1: i.e. $x = (x_1)$, $S = (s_{11})$, $f(x) = x_1 s_{11} x_1 = x_1^2 s_{11}$

$$\partial f/\partial x = 2s_{11}x_1 = 2Sx$$

Case n = 2: i.e. $x = (x_1,x_2)'$, $S = \begin{pmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{pmatrix}$,

$$\text{then } x'Sx = x_1^2 s_{11} + 2x_1x_2s_{12} + x_2^2 s_{22}$$

$$\partial f/\partial x = (\partial f/\partial x_1, \partial f/\partial x_2)' = ((2x_1s_{11}+2x_2s_{12}), (2x_1s_{12}+2x_2s_{22}))' = 2Sx.$$

General case: $f(x) = x'Sx = \Sigma_k\Sigma_j x_k s_{kj} x_j = \Sigma_k\Sigma_{j,\, j\neq k}\, x_k s_{kj} x_j + \Sigma_j s_{jj} x_j^2$,

so $\dfrac{\partial f}{\partial x_i} = \dfrac{\partial(\Sigma_k\Sigma_{j,j\neq k} x_k s_{kj} x_j + \Sigma_j s_{jj} x_j^2)}{\partial x_i} = 1\times\Sigma_{j,j\neq i} s_{ij}x_j + (\Sigma_{k,k\neq i} s_{ik}x_k)\times 1 + 2s_{ii}x_i$

$s_{ii}x_i + \Sigma_{j,j\neq i}s_{ij}x_j + (\Sigma_{k,k\neq i}s_{ik}x_k + s_{ii}x_i) = 2Sx$

Noting §1.10, clearly if A is not symmetric then the derivative of x′Ax is (A+A′)x.

6.3 Differentiation of a scalar with respect to a matrix

If X is an $m\times n$ matrix (x_{ij}) and f = f(X) is a scalar valued function of X then $\dfrac{\partial f}{\partial X}$ is ***defined*** to be the matrix

$$\frac{\partial f}{\partial X} = \begin{pmatrix} \frac{\partial f}{\partial x_{11}} & \frac{\partial f}{\partial x_{12}} & \cdots & \frac{\partial f}{\partial x_{1n}} \\ \frac{\partial f}{\partial x_{21}} & \frac{\partial f}{\partial x_{22}} & \cdots & \frac{\partial f}{\partial x_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_{m1}} & \frac{\partial f}{\partial x_{m2}} & \cdots & \frac{\partial f}{\partial x_{mn}} \end{pmatrix} = \left(\frac{\partial f}{\partial x_{ij}}\right).$$

Special care needs to be taken with this definition since the x_{ij} may not be functionally independent. For example, if the matrix X is symmetric so that $x_{ij} = x_{ji}$ then $\left(\dfrac{\partial f}{\partial x_{ij}}\right)$ may be different from the value obtained in the non-symmetric case. Symmetry is the most common situation in statistical applications where this arises but skew-symmetric matrices and also matrices of other special structures (diagonal, triangular, banded etc) need careful handling. In the cases considered below it is to be understood that there is no other functional relationship between the elements other than symmetry where that case is declared.

6.3.1 Derivative of tr(X)

If X is $n\times n$ then $tr(X) = \Sigma_k x_{kk}$ so $\left(\dfrac{\partial f}{\partial x_{ij}}\right) = 0$ if $i \neq j$ and = 1 if i = j.

Thus $\left(\dfrac{\partial f}{\partial X}\right) = I_n$

6.3.2 Derivative of a′Xa when X is not symmetric

If X is an n×n matrix and a is a vector of length n then $a'Xa = \Sigma_i\Sigma_j a_i a_j x_{ij}$

so, assuming $x_{ij} \neq x_{ji}$, $\dfrac{\partial(a'Xa)}{\partial x_{ij}} = a_i a_j$.

Thus $\dfrac{\partial(a'Xa)}{\partial X} = aa'$ provided X is not symmetric.

6.3.3 Derivative of a′Xa when X is symmetric

If X is a symmetric n×n matrix and a is a vector of length n then $a'Xa = 2\sum_{i=1}^{n}\sum_{j=1}^{i-1} a_i a_j x_{ij} + \sum_{i=1}^{n} a_i^2 x_{ij}$,

so $\dfrac{\partial(a'Xa)}{\partial x_{ij}} = 2a_i a_j$ (if $i \neq j$) and $\dfrac{\partial(a'Xa)}{\partial x_{ii}} = a_i^2$ (if $i = j$).

Thus $\dfrac{\partial(a'Xa)}{\partial X} = 2aa' - \text{diag}(aa')$ if X is symmetric.

6.3.4 Derivative of tr(XA) when X is not symmetric

If X and A are n×n matrices then $\text{tr}(XA) = \Sigma_i\Sigma_j x_{ij} a_{ji}$ so $\dfrac{\partial \text{tr}(XA)}{\partial x_{ij}} = a_{ji}$ and

thus $\dfrac{\partial \text{tr}(XA)}{\partial X} = A'$ provided X is not symmetric.

6.3.5 Derivative of tr(XA) when X is symmetric

If X and A are n×n matrices then $\text{tr}(XA) = \sum_{i=1}^{n}\sum_{j=1}^{i-1} x_{ij}(a_{ij} + a_{ji}) + \sum_{i=1}^{n} x_{ii} a_{ii}^2$

so $\dfrac{\partial \text{tr}(XA)}{\partial x_{ij}} = a_{ij} + a_{ji}$ (if $i \neq j$) and $\dfrac{\partial \text{tr}(XA)}{\partial x_{ii}} = a_{ii}$ (if $i = j$).

and thus $\dfrac{\partial \text{tr}(XA)}{\partial X} = A + A' - \text{diag}(A)$ if X is symmetric.

6.3.6 Derivative of tr(A′XA)

If X and A are m×n matrices then A′XA id m×m.

Since tr(A′XA) = tr(XAA′) (see §1.6) and since AA′ is symmetric we have $\frac{\partial \mathrm{tr}(A'XA)}{\partial X} = 2AA' - \mathrm{diag}(AA')$ or AA' according as X is symmetric or non-symmetric (by §6.3.4 and §6.3.5).

6.3.7 Derivative of |X| when X is not symmetric

If X is n×n and f = f(X) = |X| then $\frac{\partial f}{\partial X} = |X|(X^{-1})'$

First consider the case n = 2: X = (x_{ij}), f(X) = |X| = $x_{11}x_{22} - x_{12}x_{21}$.

So $\frac{\partial f}{\partial X} = \begin{pmatrix} x_{22} & -x_{21} \\ -x_{12} & x_{11} \end{pmatrix} = |X|(X^{-1})'$

6.3.7.1★★ General proof of derivative of |X| when X is not symmetric

Generally we have $|X| = \sum_{j=1}^{n} x_{jk}c_{jk}$ for any row k where c_{jk} is the cofactor of x_{jk}, see §3.1.1, so $\frac{\partial f}{\partial x_{ij}} = c_{ij}$ and thus $\frac{\partial f}{\partial X} = (c_{ij}) = (X^{\#})'$ where $X^{\#}$ is the adjoint matrix of X. Since $X^{-1} = |X|^{-1}X^{\#}$ (see §4.7⁺) we have $X^{\#} = |X|(X^{-1})'$ and thus $\frac{\partial f}{\partial X} = |X|(X^{-1})'$.

6.3.8 Derivative of |X| when X is symmetric

If X is n×n and f = f(X) = |X| then $\frac{\partial f}{\partial X} = |X|(X^{-1})'$

First consider the case n = 2: X = (x_{ij}), f(X) = |X| = $x_{11}x_{22} - x^2_{12}$.

So $\frac{\partial f}{\partial X} = \begin{pmatrix} x_{22} & -2x_{12} \\ -2x_{12} & x_{11} \end{pmatrix} = |X|\{2X^{-1} - \text{diag}(X^{-1})\}$

6.3.8.1★★ General proof of derivative of |X| when X is symmetric

Generally we have $|X| = \sum_{j=1}^{n} x_{jk}c_{jk}$ for any row k where c_{jk} is the cofactor of x_{jk}, see §3.1.1, so $\frac{\partial f}{\partial x_{ij}} = 2c_{ij}$ for $i \neq j$, and c_{ii} for $i = j$.

Thus $\frac{\partial f}{\partial X} = 2(c_{ij}) - \text{diag}(c_{ii}) = 2X^{\#} - \text{diag}((X^{\#})$ where $X^{\#}$ is the adjoint matrix of X. Since $X^{-1} = |X|^{-1}X^{\#}$ (see §4.7⁺) we have $X^{\#} = |X|(X^{-1})$ and thus $\frac{\partial f}{\partial X} = |X|\{2X^{-1} - \text{diag}(X^{-1})\}$.

6.3.9 Derivative of $|X|^r$

If f = f(X) = $|X|^r$ then $\frac{\partial f}{\partial X} = r|X|^{r-1}\frac{\partial f}{\partial X} = r|X|^{r-1}|X|(X^{-1})' = r|X|^r(X^{-1})'$, provided X is not symmetric.

When X is symmetric then clearly $\frac{\partial f}{\partial X} = |X|^r\{2X^{-1} - \text{diag}(X^{-1})\}$

6.3.10 Derivative of log(|X|)

If f = f(X) = log(|X|) then $\frac{\partial f}{\partial X} = r\,|X|^{-1}\frac{\partial f}{\partial X} = |X|^{-1}|X|(X^{-1})' = (X^{-1})'$,

provided X is not symmetric.

When X is symmetric then clearly $\frac{\partial f}{\partial X} = 2X^{-1} - \text{diag}(X^{-1})$

6.4 Differentiation of a vector with respect to a vector

If x is a vector of length n and f = f(x) is a vector function of length m

then $\frac{\partial f}{\partial x}$ is ***defined*** to be the m×n matrix

$$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix} = \left(\frac{\partial f_i}{\partial x_j}\right)$$

6.4.1 Derivative of Ax

If A is m×n then f = f(x) = Ax = $(\Sigma_j a_{1j}x_j, \Sigma_j a_{2j}x_j, \ldots, \Sigma_j a_{mj}x_j)'$,

so $$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} = A$$

6.5 Differentiation of a matrix with respect to a scalar

If the elements of a $m\times n$ matrix A are functions of a scalar x, i.e. $A = (a_{ij}) = (a_{ij}(x))$,then the derivative with respect to x is simply the matrix of derivatives of $c_{ij}(x)$ with respect to x.

6.5.1 Example $A = (1, x, x^2, x^3, \ldots, x^{n-1})'$

If $A = (1, x, x^2, x^3, \ldots, x^{n-1})'$then the derivative of A(x) with respect to x is $(0, 1, 2x, 3x^2, \ldots, (n-1)x^{n-2})$.

6.5.2 Autocorrelation matrix

$$\text{If } A(x) = \begin{pmatrix} 1 & x & x^2 & \cdots & x^{n-1} \\ x & 1 & x & x^2 & \vdots \\ x^2 & x & \ddots & & x^2 \\ \vdots & \vdots & \vdots & \ddots & x \\ x^{n-1} & \cdots & x^2 & x & 1 \end{pmatrix} \text{ then } \frac{\partial A}{\partial x} = \begin{pmatrix} 0 & 1 & x & \cdots & x^{n-2} \\ 1 & 0 & 1 & x & \vdots \\ x & 1 & \ddots & & x \\ \vdots & \vdots & \vdots & \ddots & 1 \\ x^{n-2} & \cdots & x & 1 & 0 \end{pmatrix}$$

6.6 Application to optimisation problems

6.7 Exercises 6

1.

2. if $x = (x_1, x_2, \ldots, x_n)'$ and $D = \mathrm{diag}(x)$ find $\dfrac{\partial\, \mathrm{tr}(D)}{\partial\, x}$.

3. If A, B and X are m×n matrices find $\dfrac{\partial\, \mathrm{tr}(A'XB)}{\partial X}$

4.

to be continued……

APPENDICES

These are bits from the Appendices in the course booklet for Multivariate Data Analysis. Later chapters of these notes will provide material directed towards these results, including rank, determinant and inverse of matrices.

APPENDIX 0: Background Results

A0.3 Lagrange Multipliers

Suppose $x=(x_1,...,x_n)'$. To maximize/minimize f(x) (a scalar function of x) subject to k scalar constraints $g_1(x)=0, g_2(x)=0,...,g_k(x)=0$ where $k<n$ we define $\Omega = f(x) + \sum_{j=1}^{k} \lambda_j g_j(x)$ and max/minimize Ω with respect to the n+k variables $x_1,...,x_n, \lambda_1,...,\lambda_k$.

Proof: Omitted, other than ‘by example’.

e.g. (1): $x=(x_1,x_2)'$, $f(x)=x'x=x_1^2+x_2^2$; 1 constraint $x_1+x_2=1$.

i.e. minimize $x_1^2+x_2^2$ subject to $x_1+x_2=1$.

Let $\Omega = x_1^2+x_2^2+\lambda(x_1+x_2-1)$,

$\partial\Omega/\partial x_i = 2x_i+\lambda \ (i=1,2)$, $\partial\Omega/\partial\lambda = x_1+x_2-1$. Setting these derivatives to zero yields $x_1=-\lambda/2$, $x_2=-\lambda/2$, $x_1+x_2=1$, so $\lambda=-1$ and solution is $x_{1opt}=+½$

CHECK: Substitute for x_2: $x_2=1-x_1$, $f(x)= x_1^2+(1-x_1)^2$,

$\partial f/\partial x_1 = 2x_1-2(1-x_1)$ and so $x_{1opt} = +½$ ($= x_{2opt}$).

e.g. (2): Suppose $t_1,...,t_n$ are unbiased estimates of θ with variances $\sigma_1^2,\ldots,\sigma_n^2$: to find the best linear unbiased estimate of θ. Let $\tau=\Sigma\alpha_i t_i$. We want to choose the α_i so that τ has minimum variance subject to the constraint of being unbiased. Now $E[t_i]=\theta$ all i, so $E[\tau]=\theta$, so we have the constraint $\Sigma\alpha_i=1$. Also $var(\tau)=\Sigma\alpha_i^2 var(t_i)= \Sigma\alpha_i^2\sigma_i^2$. Let $\Omega=\Sigma\alpha_i^2\sigma_i^2+\lambda(\Sigma\alpha_i-1)$:

$\partial\Omega/\partial\alpha_i = 2\alpha_i\sigma_i^2+\lambda$: $\partial\Omega/\partial\lambda = \Sigma\alpha_i - 1$.

So $\alpha_i = -\frac{1}{2}\lambda/\sigma_i^2$, so $\Sigma\frac{1}{2}\lambda/\sigma_i^2 = -1$, so $\lambda = -\frac{1}{\sum 1/2\sigma_i^2}$ and so

$\alpha_i = \frac{1}{\sigma_i^2}\left(\sum \frac{1}{\sigma_i^2}\right)^{-1}$ and the BLUE estimate of θ is $\hat{\theta} = \left(\sum t_i/\sigma_i^2\right) \Big/ \left(\sum 1/\sigma_i^2\right)$

```
> S<-matrix(c(2,1,1,1,2,1,1,1,3),3,3)
> S
     [,1] [,2] [,3]
[1,]    2    1    1
[2,]    1    2    1
[3,]    1    1    3
> eigen(S)
$values
[1] 4.414214 1.585786 1.000000

$vectors
           [,1]       [,2]          [,3]
[1,] -0.5000000 -0.5000000  7.071068e-01
[2,] -0.5000000 -0.5000000 -7.071068e-01
[3,] -0.7071068  0.7071068 -2.395816e-16

> svd(S)
$d
[1] 4.414214 1.585786 1.000000

$u
           [,1]       [,2]          [,3]
[1,] -0.5000000  0.5000000  7.071068e-01
[2,] -0.5000000  0.5000000 -7.071068e-01
[3,] -0.7071068 -0.7071068 -4.297520e-15

$v
           [,1]       [,2]          [,3]
[1,] -0.5000000  0.5000000  7.071068e-01
[2,] -0.5000000  0.5000000 -7.071068e-01
[3,] -0.7071068 -0.7071068 -7.090953e-15
eigen(t(S)%*%S)
$values
[1] 19.485281  2.514719  1.000000

$vectors
           [,1]       [,2]          [,3]
[1,] -0.5000000  0.5000000  7.071068e-01
[2,] -0.5000000  0.5000000 -7.071068e-01
[3,] -0.7071068 -0.7071068 -2.479028e-16

> S<-matrix(c(2,1,1,1,2,1,4,2,3),3,3)
> S
     [,1] [,2] [,3]
[1,]    2    1    4
[2,]    1    2    2
[3,]    1    1    3
> eigen(S)
$values
[1] 5.4494897 1.0000000 0.5505103

$vectors
          [,1]          [,2]       [,3]
[1,] 0.7157629  7.071068e-01 -0.9252592
[2,] 0.4938033 -7.071068e-01  0.2682307
[3,] 0.4938033  1.187949e-16  0.2682307

> svd(S)
$d
[1] 6.2648050 1.2687663 0.3774262
```

```
$u
           [,1]       [,2]       [,3]
[1,] -0.7249031  0.4583831 -0.5141989
[2,] -0.4442335 -0.8815778 -0.1596155
[3,] -0.5264714  0.1127186  0.8426876

$v
           [,1]       [,2]        [,3]
[1,] -0.3863665  0.1165754 -0.91494867
[2,] -0.3415655 -0.9395378  0.02452866
[3,] -0.8567694  0.3219920  0.40282413

> eigen(t(S)%*%S)
$values
[1] 39.2477815  1.6097679  0.1424505

$vectors
           [,1]       [,2]        [,3]
[1,] -0.3863665  0.1165754  0.91494867
[2,] -0.3415655 -0.9395378 -0.02452866
[3,] -0.8567694  0.3219920 -0.40282413
```